NASA/CR-1999-208979
ICASE Report No. 99-3

Priority in Process Algebras

Rance Cleaveland
State University of New York at Stony Brook, Stony Brook, New York

Gerald Lüttgen
ICASE, Hampton, Virginia

V. Natarajan
IBM Corporation, Research Triangle Park, North Carolina

Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA

Operated by Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NAS1-97046

January 1999

PRIORITY IN PROCESS ALGEBRAS*

RANCE CLEAVELAND†, GERALD LÜTTGEN‡, AND V. NATARAJAN§

Abstract. This paper surveys the semantic ramifications of extending traditional process algebras with notions of priority that allow for some transitions to be given precedence over others. These enriched formalisms allow one to model system features such as *interrupts*, *prioritized choice*, or *real-time behavior*.

Approaches to priority in process algebras can be classified according to whether the induced notion of *pre-emption* on transitions is *global* or *local* and whether priorities are *static* or *dynamic*. Early work in the area concentrated on global pre-emption and static priorities and led to formalisms for modeling interrupts and aspects of real-time, such as *maximal progress*, in *centralized* computing environments. More recent research has investigated localized notions of pre-emption in which the *distribution* of systems is taken into account, as well as dynamic priority approaches, i.e., those where priority values may change as systems evolve. The latter allows one to model behavioral phenomena such as scheduling algorithms and also enables the efficient encoding of real-time semantics.

Technically, this paper studies the different models of priorities by presenting extensions of Milner's *Calculus of Communicating Systems* (CCS) with static and dynamic priority as well as with notions of global and local pre-emption. In each case the operational semantics of CCS is modified appropriately, behavioral theories based on *strong and weak bisimulation* are given, and related approaches for different process-algebraic settings are discussed.

Key words. process algebra, priority, pre-emption, bisimulation

Subject classification. Computer Science

1. Introduction. Traditional *process algebras* [6, 37, 40, 52] provide a framework for reasoning about the *communication potential* of *concurrent* and *distributed systems*. Such theories typically consist of a simple *calculus* with a well-defined *operational semantics* [1, 63] given as *labeled transition systems*; a *behavioral equivalence* is then used to relate implementations and specifications, which are both given as terms in the calculus. In order to facilitate *compositional reasoning*, in which systems are verified on the basis of the behavior of their components, researchers have devoted great attention to the definition of *behavioral congruences*, which allow the substitution of "equals for equals" inside larger systems.

Although many case studies (see e.g. [2]) prove the utility of the process-algebraic approach to system modeling and verification, many systems in practice cannot be modeled accurately within this framework.

*This work was supported by the National Aeronautics and Space Administration under NASA Contract Nos. NAS1-97046 while the first and second authors were in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681-2199.

†Department of Computer Science, State University of New York at Stony Brook, Stony Brook, NY 11794-4400, e-mail: rance@cs.sunysb.edu. Research supported by NSF grants CCR-9257963, CCR-9402807, CCR-9505662 and INT-9603441 and AFOSR grant F49620-95-1-0508.

‡Institute for Computer Applications in Science and Engineering (ICASE), Mail Stop 403, NASA Langley Research Center, Hampton, VA 23681-2199, e-mail: luettgen@icase.edu.

§Networking Hardware Division, IBM Corporation, Research Triangle Park, NC 27709, e-mail: nataraj@raleigh.ibm.com.

One reason is that traditional process algebras focus exclusively on expressing the potential *nondeterminism* that the interplay of concurrent processes may exhibit; they do not provide any means for encoding differing levels of *urgency* among transitions that might be enabled from a given system state. Typical examples of urgency include:

- **interrupts**, where non-urgent transitions at a state are pre-empted whenever an interrupt is raised;
- **programming language constructs**, such as the `PRIALT` construct in `occam` [41], that impose an order on transitions;
- **real-time behavior** that is semantically founded on the well-known *synchrony hypothesis* [13] or *maximal progress assumption* [74]; and
- **scheduling algorithms** which also rely on the concept of pre-emption.

In each of these cases urgency provides a means for *restricting nondeterminism*. This mechanism is simply ignored in traditional process algebras. As a consequence, the resulting system models are often not faithful since they contain spurious paths that cannot be traversed by the real-world systems themselves [16, 28].

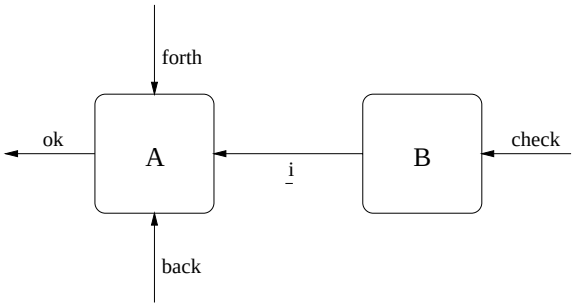

FIG. 1.1. *A simple example system*

As a simple example of the need for integrating concepts of urgency in process algebra consider the interrupt-based system depicted in Figure 1.1. It consists of two processes, A that flips back and forth between two states and B that checks if A is running properly. Whenever B receives a `check` message it requests status information from A via *interrupt port* $\underline{i}$ which in turn responds by $\overline{\texttt{ok}}$. In the absence of an indication that a communication on $\underline{i}$ is more urgent than one on `back` and `forth`, the process A can ignore a `check` request indefinitely.

1.1. Classification of Approaches to Priority. A number of approaches have been proposed for taking into account different aspects of priority [4, 12, 16, 20, 21, 22, 23, 25, 27, 28, 31, 33, 35, 42, 43, 44, 48, 49, 50, 58, 59, 65, 68, 69]. One may classify these approaches according to the following two criteria.

Static vs. dynamic priority:

In *static* approaches, transitions are assigned priority values that do not change as the system under consideration evolves. These schemes find application in the modeling of interrupts or prioritized choice constructs. In the former case, interrupts have a fixed urgency level associated with them; in the latter, priorities of transitions are fixed by the static program syntax. Almost all priority approaches to process algebra published so far deal with static priorities. The exceptions are [16, 21], which present models that allow priority values of transitions to change as systems evolve. Such *dynamic behavior* is useful in modeling scheduling approaches and real-time semantics.

Global vs. local pre-emption:

> This criterion refers to the *scope* of the priority values. In the case of centralized systems, priorities generally have a *global* scope in the sense that transitions in one process may pre-empt transitions in another. We refer to this kind of pre-emption, which has been advocated by Baeten, Bergstra, and Klop [4] and by Cleaveland and Hennessy [25] in the late eighties, as *global pre-emption*. In contrast, in a distributed system containing several sites, transitions should only be allowed to pre-empt those at the same site. This kind of pre-emption, which was first studied by Camilleri and Winskel [23] in the early nineties, is called *local pre-emption*.

Based on this classification scheme the body of this paper investigates the following different semantics for a prototypical process-algebraic language: *static/global*, *static/local*, and *dynamic/global*. The combination of dynamic priority and local pre-emption, on which research has not yet been carried out, is omitted.

Some caveats about terminology are in order here. Other process algebra researchers have used the term "pre-emption" in a setting without priorities [17]; in their usage pre-emption occurs when the execution of one transition removes the possibility of another. In our priority-oriented framework, we say that pre-emption occurs when the *presence* of one transition disables another transition. Berry [12] refers to this latter notion as *must pre-emption* and to the former as *may pre-emption*. In this article, whenever we speak of "pre-emption" we mean "must pre-emption." It should also be noted that our concept of global pre-emption and dynamic priority differs from the notion of *globally dynamic* priority found in [68]; as the distinction is somewhat technical we defer further discussion on this point to later in the article.

1.2. Summary. This paper surveys existing work on priority in process algebras. In order to focus on some of the technical issues involved with priority and pre-emption, we introduce a simple framework for their illustration. This framework extends Milner's *Calculus of Communicating Systems* (CCS) [52] and its *bisimulation*-based semantic theory by attaching priority values to actions. Although familiarity with CCS is not a prerequisite for reading this article, some knowledge of it would be advantageous since not all standard definitions and notations are re-stated here. For our language three different semantics are given: one reflecting static priorities and global pre-emption, one for static priorities and local pre-emption, and one capturing dynamic priorities and global pre-emption. The common language allows for a detailed comparison of the semantic concepts; in addition, the classification scheme presented above helps us to categorize most published approaches to priority. These have been proposed for a variety of well-known process algebras, such as the already mentioned CCS, the *Algebra of Communicating Processes* (ACP) [8], *Communicating Sequential Processes* (CSP) [40], the *Calculus of Broadcasting Systems with Priorities* (PCBS) [65], *Synchronous CCS* (SCCS) [52], and *Asynchronous Communicating Shared Resources* (ACSR) [22].

Technically, for the process algebras with static priority to be presented in this paper we develop a semantic theory based on the notion of bisimulation [52, 61]. Our aim is to carry over the standard algebraic results from CCS [52], including abstractness theorems as well as *axiomatic*, *logical*, and *algorithmic characterizations*. More precisely, we investigate both strong and weak bisimulations that are based on naive adaptations of the standard definitions as given by Milner; we especially characterize the largest congruences contained in these relations. These abstractness results indicate that the behavioral relations are semantically adequate and useful for formally reasoning about concurrent and distributed systems. Moreover, we present sound and complete axiomatizations for the obtained strong bisimulations with respect to finite processes,

i.e., those which do not contain recursion. These axiomatizations testify to the mathematical tractability of the semantic theories presented here. We also characterize the attendant notions of prioritized strong and weak bisimulations as standard bisimulations on alternative transition relations so that well-known *partition-refinement algorithms* [46, 60] for their computation become applicable. This also allows for establishing logical characterizations of the behavioral relations by adapting *Hennessy-Milner logic* [19, 52]. In the case of the dynamic priority semantics, we prove a one-to-one correspondence with traditional real-time semantics in terms of strong bisimulation. Because of this close relationship semantic theories developed for real-time process algebras can be carried over to the dynamic priority setting.

1.3. Organization. The remainder of this paper is organized as follows. The next section introduces our language, defines some formal notations used throughout the paper, and discusses some basic design decisions we have taken. Section 3 presents a semantics of the language based on static priority and global pre-emption; Section 4 then develops a semantics based on static priority and local pre-emption. A dynamic priority approach is illustrated in Section 5. Related work is referred to in each of the last three sections, while Section 6 surveys several priority approaches adopted for different process-algebraic frameworks. Section 7 contains our conclusions and suggestions for future work. The final section points to the most relevant sources of the research compiled in this article.

2. Basic Language and Notation. As mentioned above, the language considered here is an extension of Milner's CCS [52], a process algebra characterized by *handshake communication* and *interleaving semantics* for parallel composition. Syntactically, CCS includes notations for *visible actions*, which are either sends or receives on ports, and a distinguished *invisible*, or *internal* action. The semantics of CCS is then given via a transition relation that labels execution steps with actions. When a sender and receiver synchronize, the resulting action is internal. Consequently, transitions labeled by visible actions can be seen as representing only "potential" computation steps, since in order for them to occur they require a contribution from the environment. Transitions labeled by internal actions describe complete synchronizations and therefore should be viewed as "real" computation steps.

In order to capture priorities, the syntax of our language differs from CCS in that the port set exhibits a priority scheme, i.e., priorities are attached to ports. Our notion of pre-emption then stipulates that a system cannot engage in transitions labeled by actions with a given priority whenever it is able to perform a transition labeled by an *internal* action of a higher priority. In this case we say that the lower-priority transition is pre-empted by the higher-priority internal transition. In accordance with the above discussion visible actions *never* have pre-emptive power over actions of lower priority because visible actions only indicate the potential for execution. An algebraic justification of this design decision can be found in Section 3.5.

Technically, priority values are taken from some finite domain equipped with a strict order. For the sake of simplicity we use finite initial intervals $\mathcal{N}$ of the natural numbers in what follows. We adopt the convention that smaller numbers mean higher priorities; so 0 is the highest priority. Intuitively, visible actions represent potential communications that a process may be willing to engage in with its environment. Formally, let $\{\Lambda_k \mid k \in \mathcal{N}\}$ denote an $\mathcal{N}$-indexed family of countably infinite, disjoint sets of *ports*. Intuitively, Λ_k contains the ports with priority k that processes may synchronize over. Then the set of *actions* $\mathcal{A}_k$ with priority k may be defined by $\mathcal{A}_k =_{\mathrm{df}} \Lambda_k \cup \overline{\Lambda}_k \cup \{\tau_k\}$, where $\overline{\Lambda}_k =_{\mathrm{df}} \{\overline{\lambda} \mid \lambda \in \Lambda_k\}$ and $\tau_k \notin \Lambda_k$. An action $\lambda{:}k \in \Lambda_k$ may be thought of as representing the receipt of an input on port λ that has priority k, while $\overline{\lambda}{:}k \in \overline{\Lambda}_k$ constitutes the deposit of an output on λ. The invisible actions τ_k represent internal computation steps with

priority k. For better readability we write $\lambda{:}k$ if $\lambda \in \Lambda_k$ and $\tau{:}k$ for τ_k. The set of all ports Λ and the set of all actions $\mathcal{A}$ are defined by $\bigcup\{\Lambda_k \,|\, k \in \mathcal{N}\}$ and $\bigcup\{\mathcal{A}_k \,|\, k \in \mathcal{N}\}$, respectively. In what follows, we use $\alpha{:}k, \beta{:}k, \ldots$ to range over $\mathcal{A}$ and $a{:}k, b{:}k, \ldots$ to range over $\Lambda \cup \overline{\Lambda}$. We also extend $\bar{}$ to all visible actions $a{:}k$ by $\overline{\overline{a}}{:}k =_{\mathrm{df}} a{:}k$. Finally, if $L \subseteq \mathcal{A} \setminus \{\tau{:}k \,|\, k \in \mathcal{N}\}$ then $\overline{L} = \{\overline{a}{:}k \,|\, a{:}k \in L\}$. The *syntax* of our language is defined by the following BNF.

$$P \quad ::= \quad \mathbf{0} \quad | \quad x \quad | \quad \alpha{:}k.P \quad | \quad P + P \quad | \quad P \,|\, P \quad | \quad P[f] \quad | \quad P \setminus L \quad | \quad \mu x.P \ .$$

Here f is a *finite relabeling*, i.e., a mapping on $\mathcal{A}$ which satisfies $f(\tau{:}k) = \tau{:}k$ for all $k \in \mathcal{N}$, $\overline{f(\overline{a}{:}k)} = f(a{:}k)$ for all $a{:}k \in \mathcal{A} \setminus \{\tau{:}k \,|\, k \in \mathcal{N}\}$ and $|\{\alpha{:}k \,|\, f(\alpha{:}k) \neq \alpha{:}k\}| < \infty$. Moreover, a relabeling preserves priority values, i.e., for all $a{:}k \in \mathcal{A} \setminus \{\tau{:}k \,|\, k \in \mathcal{N}\}$ we have $f(a{:}k) = b{:}k$ for some $b{:}k \in \mathcal{A}_k \setminus \{\tau_k\}$. Furthermore, the *restriction set* L is a subset of $\mathcal{A} \setminus \{\tau{:}k \,|\, k \in \mathcal{N}\}$, and x is a variable taken from a set $\mathcal{V}$. Sometimes it is convenient to write $C \stackrel{\mathrm{def}}{=} P$ for $\mu C.P$ where the identifier C is interpreted as variable. We adopt the standard definitions for *sort* of a process, *free* and *bound variables*, *open* and *closed terms*, *guarded recursion*, and *contexts* [52]. We refer to closed and guarded terms as *processes* and use $P, Q, R, \ldots$ to range over the set $\mathcal{P}$ of processes. Finally, we denote syntactic equality by $\equiv$.

Although our framework allows for multi-level priority schemes we often restrict ourselves to a two-level priority framework, i.e. we choose $\mathcal{N} = \{0, 1\}$. The reason is that even in this simple setting most central semantic and technical issues regarding the introduction of priority to process algebra can be illustrated. However, we also discuss how the obtained results can be carried over to multi-level priority-schemes. In order to improve readability within the two-level priority-scheme we often write $\underline{\alpha}$ for the "prioritized" action $\alpha{:}0$, α for the "unprioritized" action $\alpha{:}1$, $\underline{A}$ for $\mathcal{A}_0$, and A for $\mathcal{A}_1$. Moreover, we let δ and γ represent elements taken from $\underline{A} \cup A$. Finally, we want to emphasize again that α and $\underline{\alpha}$ are considered to be different ports; i.e., the priority value is part of a *port* and not of an action. Thus, in a CCS-based framework only complementary actions having the *same* priority value can engage in a communication. We discuss the consequences of lifting this restriction in Section 3.7 for frameworks involving global pre-emption and in Section 4.6 for those involving local pre-emption. It should be remarked that the dynamic priority approach presented in Section 5 also differs in its interpretation of ports, actions, and priority values. Finally, our language does not provide any means for changing priority values of actions. However, we will discuss in Section 3.5 the effect of introducing additional operators to our language, called *prioritization* and *deprioritization*, which respectively increase and decrease priority values.

3. Static Priority and Global Pre-emption. In this section we introduce a semantics of our language, restricted to a two-level priority-scheme, based on static priority and global pre-emption. We refer to this language as $\mathsf{CCS}^{\mathsf{sg}}$ (CCS with static priority and global pre-emption) and develop its semantic theory along the lines mentioned in Section 1.2. The organization of this section is as follows. Section 3.1 formally introduces the operational semantics for $\mathsf{CCS}^{\mathsf{sg}}$. The following two sections show how to adapt the notions of strong bisimulation and observational congruence to $\mathsf{CCS}^{\mathsf{sg}}$, respectively. Section 3.4 applies the semantic theory to our introductory back-and-forth example. The consequences of adding prioritization and deprioritization operators to $\mathsf{CCS}^{\mathsf{sg}}$ are discussed in Section 3.5. Finally, Section 3.6 comments on the extension of $\mathsf{CCS}^{\mathsf{sg}}$ to multi-level priority-schemes whereas Section 3.7 presents our concluding remarks and related work.

3.1. Operational Semantics. The *semantics* of a process $P \in \mathcal{P}$ is given by a labeled transition system $\langle \mathcal{P}, \mathcal{A}, \longrightarrow, P \rangle$, where $\mathcal{P}$ is the set of *states*, $\mathcal{A}$ is the *alphabet*, $\longrightarrow \subseteq \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ is the transition relation formally defined to be the least relation satisfying the operational rules in Plotkin-style notation [63]

presented in Table 3.2, and P is the *start state*. We write $P \xrightarrow{\gamma} P'$ instead of $\langle P, \gamma, P' \rangle \in \longrightarrow$ and say that P *may engage in action* γ *and thereafter behave like process* P'. Moreover, we let $P \xrightarrow{\gamma}$ stand for $\exists P' \in \mathcal{P}.\, P \xrightarrow{\gamma} P'$. The presentation of the operational rules requires *prioritized initial action sets* $\underline{\mathcal{I}}(P)$ which are defined as the smallest sets satisfying the equations in Table 3.1. Intuitively, $\underline{\mathcal{I}}(P)$ denotes the set of all prioritized actions in which P can initially engage. For convenience we also write $\underline{\mathcal{II}}(P)$ for $\underline{\mathcal{I}}(P) \setminus \{\tau\}$.

TABLE 3.1
Prioritized initial action sets for CCS^{sg}

$\underline{\mathcal{I}}(\underline{\alpha}.P)$	$=$	$\{\underline{\alpha}\}$	$\underline{\mathcal{I}}(\mu x.P)$	$=$	$\underline{\mathcal{I}}(P[\mu x.P/x])$
$\underline{\mathcal{I}}(P+Q)$	$=$	$\underline{\mathcal{I}}(P) \cup \underline{\mathcal{I}}(Q)$	$\underline{\mathcal{I}}(P \,\vert\, Q)$	$=$	$\underline{\mathcal{I}}(P) \cup \underline{\mathcal{I}}(Q) \cup \{\underline{\tau} \,\vert\, \underline{\mathcal{I}}(P) \cap \overline{\underline{\mathcal{I}}(Q)} \neq \emptyset\}$
$\underline{\mathcal{I}}(P[f])$	$=$	$\{f(\underline{\alpha}) \,\vert\, \underline{\alpha} \in \underline{\mathcal{I}}(P)\}$	$\underline{\mathcal{I}}(P \setminus L)$	$=$	$\underline{\mathcal{I}}(P) \setminus (L \cup \overline{L})$

TABLE 3.2
Operational semantics for CCS^{sg}

Act	$\dfrac{-}{\underline{\alpha}.P \xrightarrow{\underline{\alpha}} P}$	$\dfrac{-}{\alpha.P \xrightarrow{\alpha} P}$
Sum1	$\dfrac{P \xrightarrow{\underline{\alpha}} P'}{P+Q \xrightarrow{\underline{\alpha}} P'}$	$\dfrac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}\ \underline{\tau} \notin \underline{\mathcal{I}}(Q)$
Sum2	$\dfrac{Q \xrightarrow{\underline{\alpha}} Q'}{P+Q \xrightarrow{\underline{\alpha}} Q'}$	$\dfrac{Q \xrightarrow{\alpha} Q'}{P+Q \xrightarrow{\alpha} Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P)$
Com1	$\dfrac{P \xrightarrow{\underline{\alpha}} P'}{P \,\vert\, Q \xrightarrow{\underline{\alpha}} P' \,\vert\, Q}$	$\dfrac{P \xrightarrow{\alpha} P'}{P \,\vert\, Q \xrightarrow{\alpha} P' \,\vert\, Q}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \,\vert\, Q)$
Com2	$\dfrac{Q \xrightarrow{\underline{\alpha}} Q'}{P \,\vert\, Q \xrightarrow{\underline{\alpha}} P \,\vert\, Q'}$	$\dfrac{Q \xrightarrow{\alpha} Q'}{P \,\vert\, Q \xrightarrow{\alpha} P \,\vert\, Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \,\vert\, Q)$
Com3	$\dfrac{P \xrightarrow{\underline{a}} P' \quad Q \xrightarrow{\underline{\overline{a}}} Q'}{P \,\vert\, Q \xrightarrow{\underline{\tau}} P' \,\vert\, Q'}$	$\dfrac{P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q'}{P \,\vert\, Q \xrightarrow{\tau} P' \,\vert\, Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \,\vert\, Q)$
Rel	$\dfrac{P \xrightarrow{\underline{\alpha}} P'}{P[f] \xrightarrow{f(\underline{\alpha})} P'[f]}$	$\dfrac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]}$
Res	$\dfrac{P \xrightarrow{\underline{\alpha}} P'}{P \setminus L \xrightarrow{\underline{\alpha}} P' \setminus L}\ \underline{\alpha} \notin L \cup \overline{L}$	$\dfrac{P \xrightarrow{\alpha} P'}{P \setminus L \xrightarrow{\alpha} P' \setminus L}\ \alpha \notin L \cup \overline{L}$
Rec	$\dfrac{P[\mu x.P/x] \xrightarrow{\underline{\alpha}} P'}{\mu x.P \xrightarrow{\underline{\alpha}} P'}$	$\dfrac{P[\mu x.P/x] \xrightarrow{\alpha} P'}{\mu x.P \xrightarrow{\alpha} P'}$

The rules in Table 3.2 capture the following operational behavior. The process $\gamma.P$ may engage in action γ and then behave like P. The *summation operator* $+$ denotes *nondeterministic choice*. The process $P+Q$ may behave like process P (Q) if Q (P) does not pre-empt an unprioritized transition by performing a prioritized internal transition. The *restriction operator* $\backslash L$ prohibits the execution of transitions labeled by actions in $L \cup \overline{L}$ and, thus, permits the *scoping* of actions. $P[f]$ behaves exactly as process P with the actions renamed with respect to f. The process $P \,|\, Q$ stands for the *parallel composition* of P and Q according to an *interleaving semantics* with *synchronized communication* on complementary actions on the same priority value resulting in the internal action $\underline{\tau}$ or τ. However, if Q (P) is capable of engaging in a prioritized internal transition, then unprioritized transitions of P (Q) are pre-empted. Finally, $\mu x.P$ denotes a *recursively defined* process that is a distinguished solution to the equation $x = P$.

3.2. Semantic Theory Based on Strong Bisimulation. The semantic theory for $\mathsf{CCS}^{\mathsf{sg}}$ is based on the notion of *bisimulation* [52, 61]. First, *strong bisimulation* [52] is adapted from CCS to our setting as follows; we refer to this relation as *prioritized strong bisimulation*.

DEFINITION 3.1 (Prioritized Strong Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is called a* prioritized strong bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$ and $\gamma \in \mathcal{A}$ the following condition holds.*

$$P \xrightarrow{\gamma} P' \text{ implies } \exists Q'.\, Q \xrightarrow{\gamma} Q' \text{ and } \langle P', Q'\rangle \in \mathcal{R} \ .$$

We write $P \simeq Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some prioritized strong bisimulation $\mathcal{R}$.

It is easy to see that $\simeq$ is an equivalence and that it is the *largest* prioritized strong bisimulation. The following result, which enables compositional reasoning, can be proved straightforwardly using standard techniques [1, 25, 72].

THEOREM 3.2. *$\simeq$ is a congruence.*

An *axiomatization* of $\simeq$ for *finite* processes, i.e., guarded and closed $\mathsf{CCS}^{\mathsf{sg}}$ terms not containing recursion, can be developed closely along the lines of [25]. We write $\vdash t = u$ if process term t can be rewritten to u using the axioms in Table 3.3, which correspond to the axioms presented in [52] except that Axiom (P) dealing with global pre-emption has been added. In *Expansion Axiom* (E) the symbol $\sum$ stands for the indexed version of $+$, where the empty sum denotes the inaction process $\mathbf{0}$. The next theorem states that our equations characterize prioritized strong bisimulation for finite $\mathsf{CCS}^{\mathsf{sg}}$ processes. Its proof can be found in [25]; it uses the technique described in [52].

THEOREM 3.3. *Let t and u be finite processes. Then $t \simeq u$ if and only if $\vdash t = u$.*

3.3. Semantic Theory Based on Weak Bisimulation. The behavioral congruence developed in the previous section is too strong for verifying systems in practice, as it requires that two equivalent terms match each other's transitions exactly, even those labeled by internal actions. In process algebra one remedies this problem by developing a semantic congruence that abstracts away from internal transitions. We start off with the definition of a *naive* prioritized weak bisimulation which is an adaptation of Milner's *observational equivalence* [52].

DEFINITION 3.4 (Naive Prioritized Weak Transition Relation).

1. $\hat{\underline{\tau}} =_{df} \hat{\tau} =_{df} \epsilon$, $\hat{\underline{a}} =_{df} \underline{a}$, *and* $\hat{a} =_{df} a$
2. $\stackrel{\epsilon}{\Longrightarrow}_{\times} =_{df} (\xrightarrow{\underline{\tau}} \cup \xrightarrow{\tau})^*$
3. $\stackrel{\gamma}{\Longrightarrow}_{\times} =_{df} \stackrel{\epsilon}{\Longrightarrow}_{\times} \circ \xrightarrow{\gamma} \circ \stackrel{\epsilon}{\Longrightarrow}_{\times}$

TABLE 3.3
Axiomatization of $\simeq$

(A1)	$t + u = u + t$	(A2)	$t + (u + v) = (t + u) + v$
(A3)	$t + t = t$	(A4)	$t + \mathbf{0} = t$

(E) Let $t = \sum_i \gamma_i.t_i$ and $u = \sum_j \delta_j.u_j$. Then
$t \,|\, u = \sum_i \gamma_i.(t_i \,|\, u) + \sum_j \delta_j.(t \,|\, u_j) +$
$\sum_{\gamma_i \equiv \overline{\delta_j}} \{\underline{\tau}.(t_i \,|\, u_j) \,|\, \gamma_i \in \underline{A}\}) + \sum_{\gamma_i \equiv \overline{\delta_j}} \{\tau.(t_i \,|\, u_j) \,|\, \gamma_i \in A\})$

(Res1)	$\mathbf{0} \setminus L = \mathbf{0}$		(Rel1)	$\mathbf{0}[f] = \mathbf{0}$
(Res2)	$(\gamma.t) \setminus L = \mathbf{0}$	$(\gamma \in L \cup \overline{L})$	(Rel2)	$(\gamma.t)[f] = f(\gamma).(t[f])$
(Res3)	$(\gamma.t) \setminus L = \gamma.(t \setminus L)$	$(\gamma \notin L \cup \overline{L})$	(Rel3)	$(t + u)[f] = t[f] + u[f]$
(Res4)	$(t + u) \setminus L = (t \setminus L) + (u \setminus L)$		(P)	$\underline{\tau}.t + \alpha.u = \underline{\tau}.t$

Observe that this transition relation abstracts from priority levels for $\stackrel{\epsilon}{\Longrightarrow}_\times$. This is in accordance with the fact that a priority value is part of an action and, thus, is unobservable for internal actions.

DEFINITION 3.5 (Naive Prioritized Weak Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* naive prioritized weak bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$, and $\gamma \in \mathcal{A}$ the following condition holds.*

$$P \stackrel{\gamma}{\longrightarrow} P' \text{ implies } \exists Q'.\, Q \stackrel{\hat{\gamma}}{\Longrightarrow}_\times Q' \text{ and } \langle P', Q'\rangle \in \mathcal{R} \ .$$

We write $P \approx_\times Q$ if there exists some naive prioritized weak bisimulation $\mathcal{R}$ such that $\langle P, Q\rangle \in \mathcal{R}$.

Naive prioritized weak bisimulation can be shown to be an equivalence. Unfortunately, $\approx_\times$ is not a congruence for $\mathsf{CCS}^{\mathsf{sg}}$ with respect to parallel composition, summation, and recursion. Whereas the compositionality defect for summation and recursion is similar to the one for CCS [52], the defect with respect to parallel composition is due to pre-emption. As an example consider the processes $P \stackrel{\text{def}}{=} a.\mathbf{0} + \underline{b}.\mathbf{0}$ and $Q \stackrel{\text{def}}{=} a.\mathbf{0} + \tau.(a.\mathbf{0} + \underline{b}.\mathbf{0})$. It is easy to see that $P \approx_\times Q$. However, when composing these processes in parallel with the process $\overline{\underline{b}}.\mathbf{0}$ then $Q \,|\, \overline{\underline{b}}.\mathbf{0} \stackrel{a}{\longrightarrow} \mathbf{0} \,|\, \overline{\underline{b}}.\mathbf{0}$ whereas $P \,|\, \overline{\underline{b}}.\mathbf{0} \stackrel{a}{\not\Longrightarrow}_\times$, i.e., $P \,|\, \overline{\underline{b}}.\mathbf{0} \not\approx_\times Q \,|\, \overline{\underline{b}}.\mathbf{0}$. This example shows that one has to be more careful when defining the prioritized weak transition relation since transitions labeled by visible actions may turn to internal transitions when composed with an environment and, thereby, may gain pre-emptive power. Consequently, a more adequate notion of weak transitions must take the potential of processes engaging in visible prioritized transitions into account.

3.3.1. Prioritized Weak Bisimulation. Despite its lack of compositionality, the above definition of $\approx_\times$ reflects an intuitive approach to abstracting from internal computation. For handling the congruence problem it is important to consider the following fact from universal algebra.

PROPOSITION 3.6. *Let $\mathcal{R}$ be an equivalence over an algebra $\Re$. The largest congruence $\mathcal{R}^+$ in $\mathcal{R}$ exists and $\mathcal{R}^+ = \{\langle P, Q\rangle \,|\, \forall\, \Re\text{-contexts } C[X].\, \langle C[P], C[Q]\rangle \in \mathcal{R}\}$, where an $\Re$-context $C[X]$ is a term in $\Re$ with one free occurrence of the variable X.*

Thus, we know that $\approx_\times$ contains a largest congruence $\approx_\times^+$ for $\mathsf{CCS}^{\mathsf{sg}}$ and devote the rest of this section to characterizing $\approx_\times^+$. We first define a new weak transition relation which takes pre-emption into account.

DEFINITION 3.7 (Prioritized Weak Transition Relation). *Let* $L \subseteq \mathcal{A} \setminus \{\underline{\tau}\}$.

(i)	$\hat{\underline{\tau}} =_{df} \underline{\epsilon}$, $\hat{\tau} =_{df} \epsilon$, $\hat{\underline{a}} =_{df} \underline{a}$, *and* $\hat{a} =_{df} a$	*(iv)*	$\stackrel{\epsilon}{\underset{L}{\Longrightarrow}} =_{df} (\xrightarrow{\underline{\tau}} \cup \stackrel{\tau}{\underset{L}{\longrightarrow}})^*$
(ii)	$P \stackrel{\alpha}{\underset{L}{\longrightarrow}} P'$ *if* $P \xrightarrow{\alpha} P'$ *and* $\underline{\underline{\mathcal{I}}}(P) \subseteq L$	*(v)*	$\stackrel{\underline{\alpha}}{\Longrightarrow} =_{df} \stackrel{\epsilon}{\Longrightarrow} \circ \xrightarrow{\underline{\alpha}} \circ \stackrel{\epsilon}{\Longrightarrow}$
(iii)	$\stackrel{\underline{\epsilon}}{\Longrightarrow} =_{df} (\xrightarrow{\underline{\tau}})^*$	*(vi)*	$\stackrel{\alpha}{\underset{L}{\Longrightarrow}} =_{df} \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} \circ \stackrel{\alpha}{\underset{L}{\longrightarrow}} \circ \stackrel{\epsilon}{\Longrightarrow}$

Intuitively, we have made the transition relation sensitive to pre-emption by introducing conditions involving prioritized initial action sets and by preserving priority levels of internal actions. In the remainder, we show that prioritized initial action sets are an adequate means for measuring pre-emption potentials. In this light, $P \stackrel{\alpha}{\underset{L}{\longrightarrow}} P'$ states that P can evolve to P' by performing the unprioritized action α if the environment does not offer any prioritized communication on some port in L.

DEFINITION 3.8 (Prioritized Weak Bisimulation). *A symmetric relation* $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ *is a* prioritized weak bisimulation *if for every* $\langle P, Q \rangle \in \mathcal{R}$, $\underline{\alpha} \in \underline{A}$, *and* $\alpha \in A$ *the following conditions hold.*

1. $\underline{\tau} \notin \underline{\mathcal{I}}(P)$ *implies* $\exists Q'.\, Q \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} Q'$, $\underline{\underline{\mathcal{I}}}(Q') \subseteq L$ *where* $L = \underline{\underline{\mathcal{I}}}(P)$, $\tau \notin \underline{\mathcal{I}}(Q')$, *and* $\langle P, Q' \rangle \in \mathcal{R}$.
2. $P \xrightarrow{\underline{\alpha}} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow} Q'$, *and* $\langle P, Q' \rangle \in \mathcal{R}$.
3. $P \xrightarrow{\alpha} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\alpha}}{\underset{L}{\Longrightarrow}} Q'$, *where* $L = \underline{\underline{\mathcal{I}}}(P)$, *and* $\langle P', Q' \rangle \in \mathcal{R}$.

We write $P \underline{\approx} Q$ *if* $\langle P, Q \rangle \in \mathcal{R}$ *for some prioritized weak bisimulation* $\mathcal{R}$.

This new version of weak bisimulation is algebraically more robust than the naive one; in fact, Condition (1) of Definition 3.8 is necessary for achieving compositionality with respect to parallel composition.

PROPOSITION 3.9. *The equivalence* $\underline{\approx}$ *is a congruence with respect to prefixing, parallel composition, relabeling, and restriction. Moreover,* $\underline{\approx}$ *is characterized as the* largest congruence *contained in* $\approx_\times$, *in the sub-algebra of* CCS^{sg} *induced by these operators and recursion.*

Although $\underline{\approx}$ is itself not a congruence, this relation provides the basis for obtaining a congruence as is made precise in the next section.

3.3.2. Prioritized Observational Congruence. The compositionality defect of $\underline{\approx}$ with respect to summation is handled in the following notion of *prioritized observational congruence*. Unfortunately, the summation fix presented in [52], which requires an initial internal transition to be matched by a nontrivial internal weak transition, is not sufficient in order to achieve a congruence based on prioritized weak bisimulation. To see why, let $D \stackrel{\text{def}}{=} \underline{\tau}.E$ and $E \stackrel{\text{def}}{=} \tau.D$. Now define $P \stackrel{\text{def}}{=} \tau.D$ and $Q \stackrel{\text{def}}{=} \underline{\tau}.E$. By Definition 3.8 we may observe $P \underline{\approx} Q$, but $P + a.\mathbf{0} \not\underline{\approx} Q + a.\mathbf{0}$ since the former can perform an a-transition whereas the latter cannot. It turns out that we have to require that observationally congruent processes must possess the same prioritized initial action sets; a requirement which is stronger than the property stated in Condition (1) of Definition 3.8.

DEFINITION 3.10. *Define* $P \underline{\approx}^{\mathrm{I}} Q$ *if for all* $\underline{\alpha} \in \underline{A}$ *and* $\alpha \in A$ *the following conditions and their symmetric counterparts hold.*

1. $\underline{\mathcal{I}}(P) \supseteq \underline{\mathcal{I}}(Q)$
2. $P \xrightarrow{\underline{\alpha}} P'$ *implies* $\exists Q'\, Q \stackrel{\underline{\alpha}}{\Longrightarrow} Q'$ *and* $P' \underline{\approx} Q'$.
3. $P \xrightarrow{\alpha} P'$ *implies* $\exists Q'.\, Q \stackrel{\alpha}{\underset{L}{\Longrightarrow}} Q'$, *where* $L = \underline{\underline{\mathcal{I}}}(P)$, *and* $P' \underline{\approx} Q'$.

The following theorem states the desired algebraic result for $\underline{\approx}^{\mathrm{I}}$.

THEOREM 3.11. $\underline{\approx}^{\mathrm{I}}$ *is the largest congruence contained in* $\approx_\times$, *i.e.,* $\underline{\approx}^{\mathrm{I}} = \approx_\times^+$.

Whereas the proof of the congruence property of $\underline{\approx}^{\mathrm{l}}$ is standard (cf., [52]), the "largest" part is proved by using the following fact from universal algebra.

PROPOSITION 3.12. *Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be equivalences over an algebra $\Re$ such that $\mathcal{R}_1^+ \subseteq \mathcal{R}_2 \subseteq \mathcal{R}_1$. Then $\mathcal{R}_1^+ = \mathcal{R}_2^+$.*

For the purposes of this section one chooses $\mathcal{R}_1 = \approx_\times$ and $\mathcal{R}_2 = \underline{\approx}$. The next theorem establishes $\mathcal{R}_2^+ = \underline{\approx}^{\mathrm{l}}$ and can be proved as a corresponding one in [52]; for details see [50].

THEOREM 3.13. *$\underline{\approx}^{\mathrm{l}}$ is the* largest *congruence contained in $\underline{\approx}$.*

In order to apply Proposition 3.12, the relation $\approx_\times^+ \subseteq \underline{\approx} \subseteq \approx_\times$ needs to be shown. The inclusion $\underline{\approx} \subseteq \approx_\times$ follows immediately from the definition of the naive prioritized weak and the prioritized weak transition relation. Thus one is left with $\approx_\times^+ \subseteq \underline{\approx}$. This inclusion turns out to be difficult to prove directly. Therefore, the auxiliary relation $\underline{\approx}_{\mathrm{a}} =_{\mathrm{df}} \{\langle P, Q\rangle \,|\, C_{PQ}[P] \approx_\times C_{PQ}[Q]\}$ is defined which lies in between $\approx_\times^+$ and $\underline{\approx}$. Here, writing S for the (finite) union of the prioritized sorts of P and Q, let $C_{PQ}[X] \stackrel{\mathrm{def}}{=} X \,|\, H_{PQ}$ and

$$H_{PQ} \stackrel{\mathrm{def}}{=} \underline{c}.\mathbf{0} + \sum_{L \subseteq \overline{S},\, \underline{b} \in S} \underline{\tau}.\begin{pmatrix} \underline{d}_{L,b}.H_{PQ} + \\ D_L + \underline{e}.H_{PQ} + \\ \overline{\underline{b}}.H_{PQ} \end{pmatrix}.$$

Moreover, D_L is defined as $\sum_{\underline{\alpha} \in L} \underline{\alpha}.\mathbf{0}$, and the actions $\underline{c}, \underline{d}_{L,\underline{b}}, \underline{e}$ for all $L \subseteq \overline{S}$ and $\underline{b} \in S$, and their complements, are supposed to be "fresh" actions, i.e., not in $S \cup \overline{S}$. By Proposition 3.6 we may conclude $\approx_\times^+ \subseteq \underline{\approx}_{\mathrm{a}}$. The other necessary inclusion $\underline{\approx}_{\mathrm{a}} \subseteq \underline{\approx}$ is established by showing that $\underline{\approx}_{\mathrm{a}}$ is a prioritized weak bisimulation; the proof details can be found in [50]. Summarizing, Theorem 3.11 is a consequence of Proposition 3.12, as is illustrated by Figure 3.1, where an arrow from relation R_1 to relation R_2 means that $R_1 \subseteq R_2$.

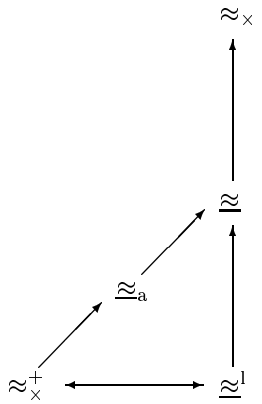

FIG. 3.1. *Situation in the proof of Theorem 3.11*

3.3.3. Operational Characterization. The aim of this section is to show how prioritized weak bisimulation can be efficiently computed by adapting standard *partition-refinement algorithms* [46, 60] developed for strong bisimulation [52]. To this end, we provide an operational characterization of prioritized weak bisimulation as strong bisimulation by introducing an *alternative prioritized weak transition relation.*

DEFINITION 3.14. *For $P, P' \in \mathcal{P}$, $\alpha \in A$, and $\underline{\alpha} \in \underline{A}$ define*

1. $\stackrel{\hat{\alpha}}{\Longrightarrow}_* =_{df} \stackrel{\hat{\alpha}}{\Longrightarrow}$ *and*
2. $P \stackrel{\hat{\alpha}}{\Longrightarrow}_* P'$ *if* $\exists P'' \in \mathcal{P}.\ \underline{\tau} \notin \underline{\mathcal{I}}(P'')$ *and* $P \stackrel{\epsilon}{\Longrightarrow} P'' \stackrel{\hat{\alpha}}{\underset{L}{\Longrightarrow}} P'$ *for* $L = \underline{\mathcal{U}}(P'')$.

Observe that the alternative prioritized weak transition relation is not parameterized by prioritized initial action sets. Its computation can be done efficiently using *dynamic programming* techniques.

DEFINITION 3.15. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is called an* alternative prioritized weak bisimulation *if for all $\langle P, Q\rangle \in \mathcal{R}$ and $\gamma \in \underline{A} \cup A$ the following condition holds.*

$$P \stackrel{\hat{\gamma}}{\Longrightarrow}_* P' \text{ implies } \exists Q'.\, Q \stackrel{\hat{\gamma}}{\Longrightarrow}_* Q' \text{ and } \langle P', Q'\rangle \in \mathcal{R} \ .$$

We write $P \underline{\approx}_* Q$ if $\langle P, Q\rangle \in R$ for some alternative prioritized weak bisimulation $\mathcal{R}$.

THEOREM 3.16 (Operational Characterization). $\underline{\approx} = \underline{\approx}_*$.

The proof is omitted since this characterization result can be established straightforwardly [50]. However, it should be mentioned that the above characterization can also be used as a basis for defining a *Hennessy-Milner logic* along the lines of [52] (see also [50]).

3.4. Example. As a simple example, we take a look at the back-and-forth system introduced in Section 1 which can be formalized in $\mathsf{CCS}^{\mathsf{sg}}$ as follows: $\mathtt{Sys} \stackrel{\text{def}}{=} (A \,|\, B) \setminus \{\underline{i}\}$ where $A \stackrel{\text{def}}{=} \mathtt{back}.A' + \underline{i}.\tau.\overline{\mathtt{ok}}.\overline{\underline{i}}.A$, $A' \stackrel{\text{def}}{=} \mathtt{forth}.A + \underline{i}.\tau.\overline{\mathtt{ok}}.\overline{\underline{i}}.A'$, and $B \stackrel{\text{def}}{=} \mathtt{check}.\overline{\underline{i}}.\underline{i}.B$. Intuitively, $\underline{i}$ is an internal *interrupt*, and thus prioritized and restricted (via $\setminus\{\underline{i}\}$), which is invoked whenever `check` is executed. Hence, in such a state the process A cannot engage in a transition labeled by `back` or `forth` according to our pre-emptive operational semantics, but must accept the communication on the prioritized port $\underline{i}$. One can think of the τ-action in the definition of process A as representing some internal activities determining the current status of the system. The $\mathsf{CCS}^{\mathsf{sg}}$ semantics of `Sys` is shown in Figure 3.2.

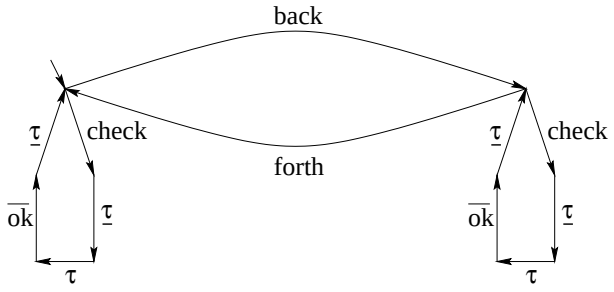

FIG. 3.2. *Semantics of* `Sys`

In the sequel, we prove that `Sys` meets its intuitive specification `Spec`, which is given by

$$\begin{aligned} \mathtt{Spec} &\stackrel{\text{def}}{=} \mathtt{back}.\mathtt{Spec}' + \mathtt{check}.\overline{\mathtt{ok}}.\mathtt{Spec} \\ \mathtt{Spec}' &\stackrel{\text{def}}{=} \mathtt{forth}.\mathtt{Spec} + \mathtt{check}.\overline{\mathtt{ok}}.\mathtt{Spec}' \ . \end{aligned}$$

First, the validity of $\mathtt{Sys} \underline{\approx} \mathtt{Spec}$ is proved by the relation presented in Table 3.4 whose symmetric closure is a prioritized weak bisimulation that contains $\langle \mathtt{Sys}, \mathtt{Spec}\rangle$.

In addition, both processes only possess visible initial actions, and their prioritized initial action sets are identical. Hence, we may conclude $\mathtt{Sys} \underline{\approx}^{\mathsf{I}} \mathtt{Spec}$. Contrast this with pure CCS where we could not deal with interrupt behavior, and we have achieved our goal.

3.5. Prioritization and Deprioritization Operators. There are several other language constructs worth considering when dealing with priority. Of particular interest are the unary operators introduced

TABLE 3.4
A relation whose symmetric closure is a prioritized weak bisimulation

$$\begin{array}{llllllllll}
\{ & \langle & \mathtt{Sys} & , & \mathtt{Spec} & \rangle, & \langle & (A' \mid B) \setminus \{\underline{i}\} & , & \mathtt{Spec}' & \rangle, \\
& \langle & (A \mid \overline{\underline{i}}.\underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec} & \rangle, & \langle & (\tau.\overline{ok}.\overline{\underline{i}}.A \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec} & \rangle, \\
& \langle & (\overline{ok}.\overline{\underline{i}}.A \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec} & \rangle, & \langle & (\overline{\underline{i}}.A \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \mathtt{Spec} & \rangle, \\
& \langle & (A' \mid \overline{\underline{i}}.\underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec}' & \rangle, & \langle & (\tau.\overline{ok}.\overline{\underline{i}}.A' \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec} & \rangle, \\
& \langle & (\overline{ok}.\overline{\underline{i}}.A' \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \overline{\mathtt{ok}}.\mathtt{Spec}' & \rangle, & \langle & (\overline{\underline{i}}.A' \mid \underline{i}.B) \setminus \{\underline{i}\} & , & \mathtt{Spec}' & \rangle & \}
\end{array}$$

by Cleaveland and Hennessy in [25] which correspond to the prioritization of a visible unprioritized action, written $\lceil a$ for $a \not\equiv \tau$, and to the deprioritization of a visible prioritized action, written $\lfloor \underline{a}$ for $\underline{a} \not\equiv \underline{\tau}$. The operational semantics of these operators is formally defined in Table 3.5. This introduction requires that (i) every prioritized port $\underline{a}$ corresponds one-to-one to an unprioritized port a, (ii) every relabeling f satisfies $f(\underline{a}) \equiv \underline{f(a)}$, and (iii) every restriction set L obeys the property "$a \in L$ if and only if $\underline{a} \in L$." Intuitively, $P\lceil a$ prioritizes all a actions which P can perform, while $P\lfloor \underline{a}$ deprioritizes all $\underline{a}$ actions in which P can engage, provided the newly deprioritized action is also available to P. Note that the notion of priority is still static and not dynamic since the prioritization and deprioritization operators are static operators. Thus, the change of priority values affects a process in its whole and is not limited to its initial behavior.

TABLE 3.5
Semantics for the prioritization and the deprioritization operator

$$\mathsf{Prio1}\ \frac{P \xrightarrow{a} P'}{P\lceil a \xrightarrow{a} P'\lceil a}\ \underline{\tau} \notin \underline{\mathcal{I}}(P) \qquad \mathsf{Prio2}\ \frac{P \xrightarrow{a} P'}{P\lceil a \xrightarrow{a} P'\lceil a}\ \underline{\tau} \in \underline{\mathcal{I}}(P) \qquad \mathsf{Prio3}\ \frac{P \xrightarrow{\gamma} P'}{P\lceil a \xrightarrow{\gamma} P'\lceil a}\ \gamma \not\equiv a$$

$$\mathsf{Deprio1}\ \frac{P \xrightarrow{\underline{a}} P'}{P\lfloor \underline{a} \xrightarrow{a} P'\lfloor \underline{a}}\ \underline{\tau} \notin \underline{\mathcal{I}}(P) \qquad \mathsf{Deprio2}\ \frac{P \xrightarrow{\underline{a}} P'}{P\lfloor \underline{a} \xrightarrow{\underline{a}} P'\lfloor \underline{a}}\ \underline{\tau} \in \underline{\mathcal{I}}(P) \qquad \mathsf{Deprio3}\ \frac{P \xrightarrow{\gamma} P'}{P\lfloor \underline{a} \xrightarrow{\gamma} P'\lfloor \underline{a}}\ \gamma \not\equiv \underline{a}$$

Including prioritization and deprioritization operators with $\mathsf{CCS}^{\mathsf{sg}}$ does not conflict with the notion of prioritized strong bisimulation; especially since it is compositional with respect to these operators [25]. The axiomatization of prioritized strong bisimulation for finite processes can also be extended to cover the new operators. The necessary additional axioms are presented in Table 3.6. Moreover, the presence of the prioritization and the deprioritization operator allows us to formally justify the design decision that only prioritized *internal* actions have pre-emptive power over unprioritized actions. For this purpose assume that (i) pre-emption is not encoded in the side conditions of the operational rules but, equivalently, in the notion of bisimulation [25] and that (iii) the naive view of pre-emption gives all prioritized actions pre-emptive power. Thus, a naive bisimulation $\sim_{\mathrm{n}}$ demands the following condition for equivalent processes $P \sim_{\mathrm{n}} Q$ and unprioritized actions $\alpha \in A$: $(P \xrightarrow{\alpha} P' \wedge\ \not\exists \underline{\beta}.\, P \xrightarrow{\underline{\beta}})$ implies $(\exists Q'.\, Q \xrightarrow{\alpha} Q' \wedge\ \not\exists \underline{\beta}.\, Q \xrightarrow{\underline{\beta}} \wedge\, P' \sim_{\mathrm{n}} Q')$, and vice versa. The condition for prioritized actions can be adopted from standard strong bisimulation. It turns out that $\sim_{\mathrm{n}}$ is not a congruence; e.g., $a.\mathbf{0} + \underline{b}.\mathbf{0} \sim_{\mathrm{n}} \underline{b}.\mathbf{0}$ but $(a.\mathbf{0} + \underline{b}.\mathbf{0}) \setminus \{\underline{b}\} \not\sim_{\mathrm{n}} (\underline{b}.\mathbf{0}) \setminus \{\underline{b}\}$ since the former process can engage in an a-transition while the latter is deadlocked. Thus, the question arises how the largest

TABLE 3.6
Axioms for the prioritization and the deprioritization operator

(Prio1)	$\mathbf{0}\lceil a$	$=$	$\mathbf{0}$	
(Prio2)	$(a.t)\lceil a$	$=$	$\underline{a}.(t\lceil a)$	
(Prio3)	$(\gamma.t)\lceil a$	$=$	$\gamma.(t\lceil a)$	$\gamma \not\equiv a$
(Prio4)	$(t+\underline{\tau}.u+\underline{b}.v)\lceil a$	$=$	$(t+\underline{\tau}.u)\lceil a+\underline{b}.(v\lceil a)$	
(Prio5)	$(t+\delta.u+\gamma.v)\lceil a$	$=$	$(t+\delta.u)\lceil a+(t+\gamma.v)\lceil a$	$\delta,\gamma \in \mathcal{A}\setminus\{\underline{\tau}\}$
(Deprio1)	$\mathbf{0}\lfloor \underline{a}$	$=$	$\mathbf{0}$	
(Deprio2)	$(\underline{a}.t)\lfloor \underline{a}$	$=$	$a.(t\lfloor \underline{a})$	
(Deprio3)	$(\gamma.t)\lfloor \underline{a}$	$=$	$\gamma.(t\lceil \underline{a})$	$\gamma \not\equiv \underline{a}$
(Deprio4)	$(t+\underline{\tau}.u+\underline{b}.v)\lfloor \underline{a}$	$=$	$(t+\underline{\tau}.u)\lfloor \underline{a}+\underline{b}.(v\lfloor \underline{a})$	
(Deprio5)	$(t+\delta.u+\gamma.v)\lfloor \underline{a}$	$=$	$(t+\delta.u)\lfloor \underline{a}+(t+\gamma.v)\lfloor \underline{a}$	$\delta,\gamma \in \mathcal{A}\setminus\{\underline{\tau}\}$

congruence $\sim_{\mathrm{n}}^{+}$ contained in $\sim_{\mathrm{n}}$ can be characterized; it turns out that $\sim_{\mathrm{n}}^{+}$ coincides with prioritized strong congruence as defined above (see [25] for a formal treatment). This shows that in the presence of pre-emption only prioritized internal actions may pre-empt unprioritized actions. However, this algebraic result is only correct if we include the deprioritization operator in our language. A non-trivial characterization of $\sim_{\mathrm{n}}^{+}$ with respect to our original language is still an open problem.

For the language extended by the prioritization and the deprioritization operator, an observational congruence together with an axiomatic characterization with respect to finite processes has been developed in [58, 59], which is briefly reviewed here. For this purpose, we need to refine the prioritized weak transition relation. First, we re-define $\stackrel{a}{\Longrightarrow}$ to $\stackrel{a}{\Longrightarrow} =_{\mathrm{df}} \stackrel{\epsilon}{\Longrightarrow} \circ \stackrel{a}{\longrightarrow} \circ \stackrel{\epsilon}{\Longrightarrow}$, i.e., a weak unprioritized a-transition consists of an a-transition that is preceded and trailed by *prioritized* internal transitions only. Moreover, we replace $\underline{\Pi}(P)$ by $\underline{\Pi}(P) \cup \Pi(P)$ in the definition of $\stackrel{\epsilon}{\Longrightarrow}$ since one has to take into account that unprioritized actions may turn to prioritized ones if they are in the scope of the prioritization operator. Finally, we write $P \stackrel{\tau}{\underset{L}{\Longrightarrow}} P'$ whenever $P \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} P'$ and $P \not\equiv P'$. Consequently, visible weak unprioritized transitions only abstract from prioritized internal actions. The reason for this restriction is that, otherwise, prioritized weak bisimulation would not be compositional with respect to the prioritization and the deprioritization operator. In contrast, the original prioritized weak transition relation allows an α-transition to be preceded by any sequence of $\underline{\tau}$- and τ-transitions (satisfying a condition on initial action sets) and only to be trailed by $\underline{\tau}$-transitions.

The notions of prioritized weak bisimulation and prioritized observational congruence are defined in [58, 59] as follows, where $P \Downarrow$ stands for $\exists P'.\, P \stackrel{\epsilon}{\Longrightarrow} P'$ and $P' \stackrel{\underline{\tau}}{\not\longrightarrow}$.

DEFINITION 3.17. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* modified prioritized weak bisimulation *if for all $\langle P, Q\rangle \in \mathcal{R}$ and $\gamma \in \mathcal{A}\setminus\{\tau\}$ the following conditions hold.*

1. *$P \Downarrow$ implies $Q \Downarrow$.*
2. *$P \stackrel{\gamma}{\longrightarrow} P'$ implies $\exists Q'.\, Q \stackrel{\hat{\gamma}}{\Longrightarrow} Q'$, and $\langle P, Q'\rangle \in \mathcal{R}$.*
3. *$P \stackrel{\tau}{\longrightarrow} P'$ implies $\exists Q'.\, Q \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} Q'$, $L = \underline{\Pi}(P) \cup \Pi(P)$, and $\langle P', Q'\rangle \in \mathcal{R}$.*

We write $P \underline{\approx}_{\mathrm{pd}} Q$ if there exists a modified prioritized weak bisimulation $\mathcal{R}$ such that $\langle P, Q\rangle \in \mathcal{R}$.

TABLE 3.7
Axioms for the τ-laws

($\underline{\tau}1$)	$\gamma.(1.t+t)$	$=$	$\gamma.t$	$1 \in \{\tau, \underline{\tau}\}$
($\underline{\tau}2$)	$\underline{\tau}.t$	$=$	$\underline{\tau}.t + t$	
($\underline{\tau}3$)	$\gamma.(t+\underline{\tau}.u)$	$=$	$\gamma.(t+\underline{\tau}.u)+\gamma.u$	
($\tau 1$)	$t+\tau.(u+\tau.v)$	$=$	$t+\tau.(u+\tau.v)+\tau.v$	$\vdash_I t \sqsubseteq_{\mathrm{i}} v$

TABLE 3.8
Axiomatization of $\sqsubseteq_{\mathrm{i}}$ (Axioms I)

(iC1)	$\underline{\alpha}.t \sqsubseteq_{\mathrm{i}} \underline{\alpha}.u$	(iC2)	$\mathbf{0} \sqsubseteq_{\mathrm{i}} \gamma.t$	$\gamma \in \mathcal{A} \setminus \{\underline{\tau}\}$	(iC3)	$\alpha.t \sqsubseteq_{\mathrm{i}} \mathbf{0}$

DEFINITION 3.18. *We define $P \underline{\approx}^{\mathrm{l}}_{\mathrm{pd}} Q$ if for all $\gamma \in \mathcal{A} \setminus \{\tau\}$ the following conditions and their symmetric counterparts hold.*

1. *$P \xrightarrow{\gamma} P'$ implies $\exists Q'\, Q \stackrel{\gamma}{\Longrightarrow} Q'$ and $P' \underline{\approx}_{\mathrm{pd}} Q'$.*
2. *$P \xrightarrow{\tau} P'$ implies $\exists Q'.\, Q \stackrel{\tau}{\underset{L}{\Longrightarrow}} Q'$, where $L = \underline{\mathit{II}}(P) \cup \mathit{II}(P)$, and $P' \underline{\approx}_{\mathrm{pd}} Q'$.*

The observational congruence $\underline{\approx}^{\mathrm{l}}_{\mathrm{pd}}$ possesses nice algebraic properties for our language extended by the prioritization and the deprioritization operator, including a largest congruence result similar to Theorem 3.11 and a sound and complete axiomatization with respect to finite processes. For the latter, the axiomatization for prioritized strong bisimulation is augmented with suitable τ-laws as shown in Table 3.7 (cf. [52]). The relation $\sqsubseteq_{\mathrm{i}}$, occurring in the side condition of Axiom ($\tau 1$), is the pre-congruence on finite processes generated from the axioms presented in Table 3.8 using the laws of inequational reasoning; we write $\vdash_I t \sqsubseteq_{\mathrm{i}} u$ if t can be related to u by Axioms (iC1), (iC2), and (iC3). Intuitively, $\vdash_I t \sqsubseteq_{\mathrm{i}} u$ holds, whenever (i) $\tau \in \underline{\mathit{I}}(t)$ if and only if $\tau \in \underline{\mathit{I}}(u)$ and (ii) $\underline{\mathit{II}}(t) \subseteq \underline{\mathit{II}}(u)$.

Finally, it should be noted that applications underline the importance of the additional freedom of abstracting from internal transitions gained by leaving out the prioritization and the deprioritization operator. In fact, the observational congruence $\underline{\approx}^{\mathrm{l}}_{\mathrm{pd}}$ does not relate the processes `Sys` and `Spec` of our back-and-forth example. This is due to the presence of the unprioritized internal action in `Sys`.

3.6. Extension to Multi-level Priority Schemes. We now remark on the extension of $\mathsf{CCS}^{\mathsf{sg}}$ to a multi-level priority-scheme. To do so, we first alter the definition of prioritized initial action sets to capture the priority-level of actions; i.e., we define sets $\mathrm{I}^k(P)$ for processes P with respect to priority value k. This can be done as shown in Table 3.9.

Using this definition of initial action sets and the convention $\tau \notin \mathrm{I}^{<k}(P)$ if $\nexists l < k.\ \tau{:}l \in \mathrm{I}^l(P)$ the operational semantics can be re-stated as follows, as exemplary shown for Rule (Com3).

$$\mathsf{Com3}\ \frac{P \xrightarrow{a:k} P' \quad Q \xrightarrow{\overline{a}:k} Q'}{P \,|\, Q \xrightarrow{\tau:k} P' \,|\, Q'}\ \tau \notin \mathrm{I}^{<k}(P \,|\, Q)$$

Observe that the sets $\mathrm{I}^k(P)$ may contain actions in which P cannot initially engage, since their definition does

TABLE 3.9
Potential initial action sets for CCS^{sg}

$$
\begin{array}{lcllcl}
\mathrm{I}^k(\alpha{:}l.P) & = & \{\alpha{:}l \mid l = k\} & \mathrm{I}^k(P[f]) & = & \{f(\alpha{:}l) \mid \alpha{:}l \in \mathrm{I}^k(P)\} \\
\mathrm{I}^k(\mu x.P) & = & \mathrm{I}^k(P[\mu x.P/x]) & \mathrm{I}^k(P{+}Q) & = & \mathrm{I}^k(P) \cup \mathrm{I}^k(Q) \\
\mathrm{I}^k(P \setminus L) & = & \mathrm{I}^k(P) \setminus (L \cup \overline{L}) & \mathrm{I}^k(P \mid Q) & = & \mathrm{I}^k(P) \cup \mathrm{I}^k(Q) \cup \{\tau{:}k \mid \mathrm{I}^k(P) \cap \overline{\mathrm{I}^k(Q)} \neq \emptyset\}
\end{array}
$$

not consider pre-emption. In fact, the set of actions with priority value k in which P can indeed initially engage is given by $\mathcal{I}^k(P) = \{\alpha{:}k \in \mathrm{I}^k(P) \mid \tau{:}l \notin \mathrm{I}^l(P) \text{ for all } l < k\}$. However, it is easy to show that $\tau \notin \mathrm{I}^{<k}(P)$ if and only if $\tau \notin \mathcal{I}^{<k}(P)$ [50]. Thus, the side condition of Rule (Com3) captures our intuition that $P \mid Q$ cannot engage in a more urgent internal transition.

TABLE 3.10
Prioritized weak transition relation

$$
\begin{array}{ll}
\stackrel{\epsilon:0}{\Longrightarrow} =_{\mathrm{df}} (\stackrel{\tau:0}{\longrightarrow})^* & P \stackrel{\alpha:k}{\underset{L}{\longrightarrow}} P' \text{ if } P \stackrel{\alpha:k}{\longrightarrow} P' \text{ and } \mathcal{I}^l(P) \subseteq L \text{ for all } l < k \\
\stackrel{\epsilon:k}{\underset{L}{\Longrightarrow}} =_{\mathrm{df}} (\{ \stackrel{\tau:l}{\underset{L}{\longrightarrow}} \mid l \leq k\})^* & \stackrel{\alpha:k}{\underset{L}{\Longrightarrow}} =_{\mathrm{df}} \stackrel{\epsilon:k}{\underset{L}{\Longrightarrow}} \circ \stackrel{\alpha:k}{\underset{L}{\longrightarrow}} \circ \stackrel{\epsilon:0}{\Longrightarrow}
\end{array}
$$

The re-development of the bisimulation-based semantic theory proceeds along the lines of the above sections and does not raise any new semantic issues. For example, the notion of prioritized observational congruence is defined as follows [50], where (i) the prioritized weak transition relation is given by the rules in Table 3.10, (ii) $\mathcal{I}\!\!\mathcal{I}^k(P) =_{\mathrm{df}} \mathcal{I}^k(P) \setminus \{\tau{:}k\}$, (iii) $\underline{\approx}_{\mathsf{ml}}$ is the adaption of prioritized weak bisimulation to a multi-level priority-scheme, (iv) $\mathcal{I}(P) =_{\mathrm{df}} \bigcup\{\mathcal{I}^k(P) \mid k \in \mathcal{N}\}$, and (v) $\mathcal{I}\!\!\mathcal{I}^{<k}(P) =_{\mathrm{df}} \mathcal{I}^{<k}(P) \setminus \{\tau{:}l \mid l < k\}$.

DEFINITION 3.19. *Processes P and Q are* prioritized observational congruent *if for all actions $\alpha{:}k$ the following conditions and their symmetric counterparts hold.*

1. $\mathcal{I}(P) \supseteq \mathcal{I}(Q)$
2. $P \stackrel{\alpha:k}{\longrightarrow} P'$ *implies* $\exists Q'.\, Q \stackrel{\alpha:k}{\underset{L}{\Longrightarrow}} Q'$, *where* $L = \mathcal{I}\!\!\mathcal{I}^{<k}(P)$, *and* $P' \underline{\approx}_{ml} Q'$

Details of the extension of CCS^{sg} to a multi-level priority-scheme can be found in [50].

3.7. Concluding Remarks and Related Work. We conclude by first commenting on the design decision that priority values are considered to be part of ports, which implies that only complementary actions having the same priority can synchronize. Lifting this design decision by allowing $a{:}k$ and $\overline{a}{:}l$, where $k \neq l$, to synchronize leads to the question of which priority value to assign to the resulting τ. One can imagine several obvious choices for this function, e.g., maximum or minimum. In addition, [33, 35] recommend using the sum of the priority values of the actions involved. Unfortunately, while a specific function may be suitable for certain examples, it is difficult to motivate for general applications. In the next section, we will see that such a function is superfluous when dealing with *local pre-emption*.

Regarding related work, Gerber and Lee have developed a real-time process algebra, the *Calculus of Communicating Shared Resources* (CCSR) [32], that explicitly takes into account the availability of system resources. Semantically, synchronizations between processes are modeled in an interleaving fashion using

instantaneous transitions, whereas the access of resources is truly concurrent and consumes time. In CCSR a priority structure may be defined over resources in order to indicate their importance; e.g., it can be used to ensure that deadlines are met. The underlying concept of priority is similar to that of $\mathsf{CCS}^{\mathsf{sg}}$ in that priorities are static and pre-emption is global. In [33] a resource-based prioritized (strong) bisimulation for CCSR together with a congruence result and axiomatizations with respect to several classes of processes [20] are given.

Prasad has also extended his *Calculus of Broadcasting Systems* (CBS) [64] for dealing with a notion of static priority [65]. He refers to the priority calculus as PCBS. For PCBS nice semantic theories based on Milner's strong and weak bisimulation have been developed along with congruence proofs. Remarkably, these theories do not suffer from the technical subtleties which have been encountered for $\mathsf{CCS}^{\mathsf{sg}}$, although the concept of pre-emption is basically the same. The reason is that PCBS uses a much simpler model for communication that is based on the principle of *broadcasting*. In this setting, priority values are only attached to output actions, which cannot be restricted or hidden as in traditional process algebras. Finally, it should be mentioned that PCBS contains an operator, called *translate*, which allows for the prioritization and the deprioritization of actions.

4. Static Priority and Local Pre-emption. This section provides a new semantics for our language, subsequently referred to as $\mathsf{CCS}^{\mathsf{sl}}$ (CCS with static priority and local pre-emption), which is distinguished from the one developed in the previous section by the design decision that it only allows actions to pre-empt others at the same "location" and therefore captures a notion of *localized precedence*. This constraint reflects an essential intuition about distributed systems, namely, that the execution of a process on one processor should not affect the behavior of a process on another processor unless the designer explicitly builds an interaction, e.g., a synchronization, between them.

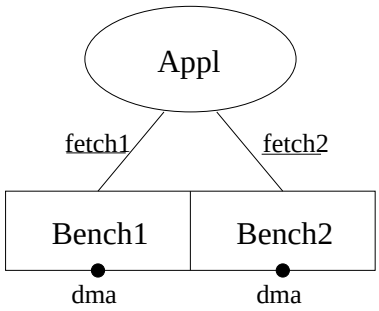

FIG. 4.1. *Example system*

The following example demonstrates the necessity to consider locations when reasoning about priority within distributed systems. The example system consists of an application that manipulates data from two memory benches (cf. Figure 4.1). In order to improve the efficiency in the computer system, each bench, `Bench1` and `Bench2`, is connected to a direct-memory-access (DMA) controller. To overcome the low speed of most memory modules, the application `Appl` works alternately with each memory bench. We model `Appl` in $\mathsf{CCS}^{\mathsf{sl}}$ by $\texttt{Appl} \stackrel{\text{def}}{=} \underline{\texttt{fetch1}}.\underline{\texttt{fetch2}}.\texttt{Appl}$. Each memory bench, `Bench1` and `Bench2`, is continuously able to serve the application or to allow the external DMA controller to access the memory via the channel `dma`. However, if a memory bench has to decide between both activities, then it chooses the former since the progress of the application is considered more important. Consequently, we define $\texttt{Bench1} \stackrel{\text{def}}{=} \underline{\texttt{fetch1}}.\texttt{Bench1} + \texttt{dma}.\texttt{Bench1}$

and $\texttt{Bench2} \stackrel{\text{def}}{=} \underline{\texttt{fetch2}}.\texttt{Bench2} + \texttt{dma}.\texttt{Bench2}$. The overall system Sys is given by

$$\texttt{Sys} \stackrel{\text{def}}{=} (\texttt{Appl} \,|\, \texttt{Bench1} \,|\, \texttt{Bench2}) \setminus \{\underline{\texttt{fetch1}}, \underline{\texttt{fetch2}}\} \ .$$

Since the application uses the memory cells alternately, the DMA is expected to be allowed to access the memory bench which is currently not serving the application. However, using the approach to priority involving global pre-emption presented in Section 3 all dma-transitions in the labeled transition system of Sys are pre-empted since the application can indefinitely engage in a prioritized communication, i.e., direct-memory-access is never granted.

Generally speaking, one would expect that priorities at different sites of a distributed system do not influence the behavior of each other, i.e., priorities at different sites are supposed to be incomparable. The semantics given in Section 3 does not permit this distinction to be made; the net effect is that some computations that one would expect to find in a distributed system are improperly suppressed. It has been proposed to remedy this shortcoming regarding distributed systems by introducing *local pre-emption* [23, 27].

The remainder of this section is organized as follows. The next section introduces a notion of locations that is used in Section 4.2 for the definition of the operational semantics of $\mathsf{CCS}^{\mathsf{s|}}$ with a two-level priority-scheme. Sections 4.3 and 4.4 develop the semantic theories based on strong and weak bisimulation, respectively, while Section 4.5 re-considers the direct-memory-access example presented above. The consequences of lifting some design decisions in $\mathsf{CCS}^{\mathsf{s|}}$ are discussed in Section 4.6. After extending $\mathsf{CCS}^{\mathsf{s|}}$ to a multi-level priority-scheme in Section 4.7 and presenting another approach to priority taken from [23] in Section 4.8, a formal comparison of the two approaches is given in Section 4.9. Finally, Section 4.10 concludes with some additional remarks and comments on related work.

4.1. Locations. We now introduce the notion of *location*, which will be used in the next section in the operational semantics for $\mathsf{CCS}^{\mathsf{s|}}$ as a basis for deciding when one transition pre-empts another. Intuitively, a location represents the "address(es)" of subterm(s) inside a larger term; when a system performs an action, $\mathsf{CCS}^{\mathsf{s|}}$ semantics will also note the location of the subterm(s) that "generate(s)" this action. Observe that because of the potential for synchronization more than one subterm may be involved in an action. The account of locations closely follows that of [27, 56].

Formally, let $\mathcal{A}_{\text{addr}} =_{\text{df}} \{L, R, l, r\}$ be the *address alphabet*, and let $\bullet$ be a special symbol not in $\mathcal{A}_{\text{addr}}$. Then, $\mathcal{A}ddr =_{\text{df}} \{\bullet s \mid s \in \mathcal{A}^*_{\text{addr}}\}$ represents the set of (process) *addresses* ranged over by v, w. Intuitively, an element of $\mathcal{A}ddr$ represents the address of a subterm, with $\bullet$ denoting the current term, l (r) representing the left (right) subterm of $+$, and L (R) the left (right) subterm of $|$. For example, in the process $(a.\mathbf{0} \,|\, b.\mathbf{0}) + c.\mathbf{0}$, the address of $a.\mathbf{0}$ is $\bullet Ll$, of $b.\mathbf{0}$ is $\bullet Rl$, and of $c.\mathbf{0}$ is $\bullet r$. If $\bullet s_1$ and $\bullet s_2$ are addresses, then we write $\bullet s_1 \cdot \bullet s_2 = \bullet s_1 s_2$ to represent address concatenation (where $s_1 s_2$ represents the usual concatenation of elements in $\mathcal{A}^*_{\text{addr}}$). Further, if $V \subseteq \mathcal{A}ddr$ and $\zeta \in \mathcal{A}_{\text{addr}}$, then we write $V \cdot \zeta$ for $\{v \cdot \zeta \mid v \in V\}$. Occasionally, we omit $\bullet$ from addresses.

As mentioned in the previous section, we want to adopt the view that processes at different sides of the parallel composition operator are logically – not necessarily physically – executed on different processors. Thus, priorities on different sides of the parallel composition operator are distributed and, therefore, should be incomparable. However, priorities on different sides of the summation operator should be comparable since argument processes of summation are logically scheduled on the same processor. This intuition is formalized in the following *comparability relation* on addresses which is adapted from [35].

TABLE 4.1
Distributed prioritized initial action sets for $\mathsf{CCS}^{\mathsf{sl}}$

$\underline{\mathcal{I}}_m(\mu x.P)$	$=$	$\underline{\mathcal{I}}_m(P[\mu x.P/x])$	$\underline{\mathcal{I}}_\bullet(\underline{\alpha}.P)$	$=$	$\{\underline{\alpha}\}$
$\underline{\mathcal{I}}_{m\cdot l}(P+Q)$	$=$	$\underline{\mathcal{I}}_m(P)$	$\underline{\mathcal{I}}_{n\cdot r}(P+Q)$	$=$	$\underline{\mathcal{I}}_n(Q)$
$\underline{\mathcal{I}}_m(P[f])$	$=$	$\{f(\underline{\alpha}) \mid \underline{\alpha} \in \underline{\mathcal{I}}_m(P)\}$	$\underline{\mathcal{I}}_m(P \setminus L)$	$=$	$\underline{\mathcal{I}}_m(P) \setminus (L \cup \overline{L})$
$\underline{\mathcal{I}}_{m\cdot L}(P\mid Q)$	$=$	$\underline{\mathcal{I}}_m(P)$	$\underline{\mathcal{I}}_{n\cdot R}(P\mid Q)$	$=$	$\underline{\mathcal{I}}_n(Q)$
$\underline{\mathcal{I}}_{\langle m\cdot L, n\cdot R\rangle}(P\mid Q)$	$=$	$\{\underline{\tau} \mid \underline{\mathcal{I}}_m(P) \cap \overline{\underline{\mathcal{I}}_n(Q)} \neq \emptyset\}$			

DEFINITION 4.1 (Comparability Relation). *The comparability relation* $\bowtie$ *on addresses is the smallest reflexive and symmetric subset of* $\mathcal{A}ddr \times \mathcal{A}ddr$ *such that for all* $v, w \in \mathcal{A}ddr$*:*

1. $\langle v \cdot l, w \cdot r\rangle \in \bowtie$*, and*
2. $\langle v, w\rangle \in \bowtie$ *implies* $\langle v \cdot \zeta, w \cdot \zeta\rangle \in \bowtie$ *for* $\zeta \in \mathcal{A}_{addr}$*.*

In the sequel we write $v \bowtie w$ instead of $\langle v, w\rangle \in \bowtie$. If $v \in \mathcal{A}ddr$ then we use $[v]$ to denote the set $\{w \in \mathcal{A}ddr \mid v \bowtie w\}$. Observe that the comparability relation is not transitive, e.g., we have $Ll \bowtie r$ and $r \bowtie Rl$, but $Ll \not\bowtie Rl$, since $L \not\bowtie R$.

We may now define the set $\mathcal{L}oc$ of (transition) *locations* as $\mathcal{A}ddr \cup (\mathcal{A}ddr \times \mathcal{A}ddr)$. Intuitively, a transition location records the addresses of the components in a term that participate in the execution of a given action. In our language, transitions are performed by single processes or pairs of processes (in the case of a synchronization). We define $\langle v, w\rangle \cdot \zeta =_{\mathrm{df}} \langle v \cdot \zeta, w \cdot \zeta\rangle$ and $[\langle v, w\rangle] =_{\mathrm{df}} [v] \cup [w]$ where $v, w \in \mathcal{A}ddr$ and $\zeta \in \mathcal{A}_{\mathrm{addr}}$. We use $m, n, o, \ldots$ to range over $\mathcal{L}oc$ in what follows.

4.2. Operational Semantics. The *operational semantics* of a $\mathsf{CCS}^{\mathsf{sl}}$ process P is given by a labeled transition system. The transition relation $\longrightarrow \subseteq \mathcal{P} \times (\mathcal{L}oc \times \mathcal{A}) \times \mathcal{P}$ with respect to unprioritized actions is defined in Table 4.2 using Plotkin-style operational rules [63] whereas for prioritized actions the same rules as for $\mathsf{CCS}^{\mathsf{sg}}$ apply (see Table 3.2). We write $P \xrightarrow{m,\alpha} P'$ if $\langle P, \langle m, \alpha\rangle, P'\rangle \in \longrightarrow$ and say that P *may engage in action* α *offered from location* m *and thereafter behave like process* P'. Note that prioritized transitions do not need to be labeled with locations since they can never be pre-empted.

The presentation of the operational rules requires *distributed prioritized initial action sets*, which are defined as the least sets satisfying the equations in Table 4.1. Intuitively, $\underline{\mathcal{I}}_m(P)$ denotes the set of all prioritized initial actions of P from location m. Note that these sets are either empty or contain exactly one initial action. $\underline{\mathcal{I}}_m(P) = \emptyset$ means that either m is not a location of P or P is incapable of performing a prioritized action at location m. Additionally, let us denote the set $\bigcup\{\underline{\mathcal{I}}_m(P) \mid m \in M\}$ of all distributed prioritized initial actions of P from locations $M \subseteq \mathcal{L}oc$ by $\underline{\mathcal{I}}_M(P)$ and the set $\underline{\mathcal{I}}_{\mathcal{L}oc}(P)$ of all distributed prioritized initial actions of P by $\underline{\mathcal{I}}(P)$. We also define analogue sets restricted to visible actions: $\underline{\mathcal{I}\!\!\mathcal{I}}_M(P) =_{\mathrm{df}} \underline{\mathcal{I}}_M(P) \setminus \{\underline{\tau}\}$ and $\underline{\mathcal{I}\!\!\mathcal{I}}(P) =_{\mathrm{df}} \underline{\mathcal{I}}(P) \setminus \{\underline{\tau}\}$, respectively.

The side conditions of the operational rules guarantee that a process does not perform an unprioritized action if it can engage in a prioritized synchronization or internal computation, i.e., a $\underline{\tau}$-transition, from a comparable location. In contrast to the global notion of pre-emption defined in Section 3, the local notion here is much weaker since $\underline{\mathcal{I}\!\!\mathcal{I}}_{[m]}(P) \subseteq \underline{\mathcal{I}\!\!\mathcal{I}}(P)$ for all $m \in \mathcal{L}oc$ and $P \in \mathcal{P}$. In other words, local pre-emption

TABLE 4.2
Operational semantics for $\mathsf{CCS}^{\mathsf{sl}}$

$$\mathsf{Act}\quad \frac{-}{\alpha.P \xrightarrow{\bullet,\alpha} P} \qquad \mathsf{Sum1}\quad \frac{P \xrightarrow{m,\alpha} P'}{P+Q \xrightarrow{m\cdot l,\alpha} P'}\ \underline{\tau} \notin \underline{\mathcal{I}}(Q)$$

$$\mathsf{Rel}\quad \frac{P \xrightarrow{m,\alpha} P'}{P[f] \xrightarrow{m,f(\alpha)} P'[f]} \qquad \mathsf{Sum2}\quad \frac{Q \xrightarrow{n,\alpha} Q'}{P+Q \xrightarrow{n\cdot r,\alpha} Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P)$$

$$\mathsf{Res}\quad \frac{P \xrightarrow{m,\alpha} P'}{P \setminus L \xrightarrow{m,\alpha} P' \setminus L}\ \alpha \notin L \cup \overline{L} \qquad \mathsf{Com1}\quad \frac{P \xrightarrow{m,\alpha} P'}{P|Q \xrightarrow{m\cdot L,\alpha} P'|Q}\ \underline{\mathcal{I}}_{[m]}(P) \cap \overline{\underline{\mathcal{I}}(Q)} = \emptyset$$

$$\mathsf{Rec}\quad \frac{P[\mu x.P/x] \xrightarrow{m,\alpha} P'}{\mu x.P \xrightarrow{m,\alpha} P'} \qquad \mathsf{Com2}\quad \frac{Q \xrightarrow{n,\alpha} Q'}{P|Q \xrightarrow{n\cdot R,\alpha} P|Q'}\ \underline{\mathcal{I}}_{[n]}(Q) \cap \overline{\underline{\mathcal{I}}(P)} = \emptyset$$

$$\mathsf{Com3}\quad \frac{P \xrightarrow{m,a} P' \quad Q \xrightarrow{n,\overline{a}} Q'}{P|Q \xrightarrow{\langle m\cdot L, n\cdot R\rangle,\tau} P'|Q'}\ \begin{array}{l} \underline{\mathcal{I}}_{[m]}(P) \cap \overline{\underline{\mathcal{I}}(Q)} = \emptyset \wedge \\ \underline{\mathcal{I}}_{[n]}(Q) \cap \overline{\underline{\mathcal{I}}(P)} = \emptyset \end{array}$$

does not pre-empt as many transitions as global pre-emption does. The difference between $\mathsf{CCS}^{\mathsf{sl}}$ and $\mathsf{CCS}^{\mathsf{sg}}$ semantics arises by the side conditions of the rules for parallel composition with respect to unprioritized transitions. Since locations on different sides of a parallel operator are incomparable, $\underline{\tau}$'s arising from a location of P (Q) cannot pre-empt the execution of a transition, even an unprioritized one, of Q (P). Only if P (Q) engages in a prioritized synchronization with Q (P) can unprioritized actions from a comparable location of P (Q) be pre-empted.

4.3. Semantic Theory Based on Strong Bisimulation. Just as in Section 3, we present an equivalence relation for $\mathsf{CCS}^{\mathsf{sl}}$ processes that is based on bisimulation [61]. Our aim is to characterize the largest congruence contained in the "naive" adaptation of strong bisimulation [52] to our framework obtained by ignoring location information.

DEFINITION 4.2 (Naive Distributed Prioritized Strong Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is called* naive distributed prioritized strong bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$ and $\gamma \in \mathcal{A}$ the following condition holds.*

$$P \xrightarrow{\gamma} P' \text{ implies } \exists Q'.\, Q \xrightarrow{\gamma} Q' \text{ and } \langle P', Q'\rangle \in \mathcal{R}\ .$$

We write $P \simeq Q$ if there exists a naive distributed prioritized strong bisimulation $\mathcal{R}$ such that $\langle P, Q\rangle \in \mathcal{R}$.

Although $\simeq$ is an equivalence, it is unfortunately – in contrast to the situation in Section 3.2 – *not* a congruence. The lack of compositionality is demonstrated by the following example, which embodies the traditional view that "*parallelism = nondeterminism.*" We have $a.\underline{b}.\mathbf{0} + \underline{b}.a.\mathbf{0} \simeq a.\mathbf{0} \,|\, \underline{b}.\mathbf{0}$ but $(a.\underline{b}.\mathbf{0} + \underline{b}.a.\mathbf{0}) \,|\, \overline{\underline{b}}.\mathbf{0} \not\simeq (a.\mathbf{0} \,|\, \underline{b}.\mathbf{0}) \,|\, \overline{\underline{b}}.\mathbf{0}$, since the latter process can perform an a-transition while the corresponding a-transition of the former is pre-empted because the right process in the summation can engage in a prioritized communication. The above observation is not surprising since the distribution of processes influences the pre-emption of transitions and, consequently, the bisimulation. However, we know by Proposition 3.6 that $\simeq$ includes a largest congruence $\simeq^{+}$ for $\mathsf{CCS}^{\mathsf{sl}}$.

TABLE 4.3
Axiomatization of $\simeq^{\mathsf{l}}$ (Axioms E)

(iA1)	$t \oplus u = u \oplus t$	(iA2)	$t \oplus (u \oplus v) = (t \oplus u) \oplus v$
(iA3)	$t \oplus t = t$	(iA4)	$t \oplus \mathbf{0} = t$

(E) $t \equiv \bigoplus_i \sum_j \gamma_{ij}.t_{ij}$ and $u \equiv \bigoplus_k \sum_l \delta_{kl}.u_{kl}$ implies $t \,|\, u =$

$$\begin{aligned}
&\bigoplus_i \textstyle\sum_j(\gamma_{ij}.(t_{ij} \,|\, u) + \sum_k \sum_l\{\tau.(t_{ij} \,|\, u_{kl}) \,|\, \gamma_{ij} \equiv \overline{\delta}_{kl},\ \gamma_{ij},\delta_{kl} \in A\} \\
&\qquad + \textstyle\sum_k \sum_l\{\underline{\tau}.(t_{ij} \,|\, u_{kl}) \,|\, \gamma_{ij} \equiv \overline{\delta}_{kl},\ \gamma_{ij},\delta_{kl} \in \underline{A}\}) \oplus \\
&\bigoplus_k \textstyle\sum_l(\delta_{kl}.(t \,|\, u_{kl}) + \sum_i \sum_j\{\tau.(t_{ij} \,|\, u_{kl}) \,|\, \gamma_{ij} \equiv \overline{\delta}_{kl},\ \gamma_{ij},\delta_{kl} \in A\} \\
&\qquad + \textstyle\sum_i \sum_j\{\underline{\tau}.(t_{ij} \,|\, u_{kl}) \,|\, \gamma_{ij} \equiv \overline{\delta}_{kl},\ \gamma_{ij},\delta_{kl} \in \underline{A}\})
\end{aligned}$$

(iRes4)	$(t \oplus u) \setminus L = (t \setminus L) \oplus (u \setminus L)$	(iRel3)	$(t \oplus u)[f] = t[f] \oplus u[f]$

4.3.1. Distributed Prioritized Strong Bisimulation. In the remainder, we develop a characterization of $\simeq^{+}$. To do so we need to take *local* pre-emption into account.

DEFINITION 4.3. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* distributed prioritized strong bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$, $\alpha \in A$, $\underline{\alpha} \in \underline{A}$, and $m \in \mathcal{L}oc$ the following conditions hold.*

1. *$P \xrightarrow{\underline{\alpha}} P'$ implies $\exists Q'.\, Q \xrightarrow{\underline{\alpha}} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.*
2. *$P \xrightarrow{m,\alpha} P'$ implies $\exists Q', n.\, Q \xrightarrow{n,\alpha} Q'$, $\underline{\mathit{\Pi}}_{[n]}(Q) \subseteq \underline{\mathit{\Pi}}_{[m]}(P)$, and $\langle P', Q'\rangle \in \mathcal{R}$.*

We write $P \simeq^{\mathsf{l}} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some distributed prioritized strong bisimulation $\mathcal{R}$.

Intuitively, the distributed prioritized initial action set of a process with respect to some location is a measure of the pre-emptive power of the process relative to that location. Thus, the second condition of Definition 4.3 states that an unprioritized action α from some location m of the process P must be matched by the same action from some location n of Q and that the pre-emptive power of Q relative to n is at most as strong as the pre-emptive power of P relative to m. The following theorem is the main result of this section.

THEOREM 4.4. *$\simeq^{\mathsf{l}}$ is the* largest *congruence contained in $\simeq$.*

We refer for the proof to [50]. The context needed in the largest congruence proof is similar to the one used in Section 3.3.

4.3.2. Axiomatic Characterization. In this section we present an axiomatization of $\simeq^{\mathsf{l}}$ with respect to *finite* processes for which we introduce a new binary summation operator $\oplus$ to the process algebra $\mathsf{CCS}^{\mathsf{sl}}$. This operator is called *distributed summation* and is needed for giving an *Expansion Axiom* (cf. Axiom (E) in Table 4.3). Its operational semantics is defined below and differs form the nondeterministic choice operator $+$ in that a location from its left argument is never comparable to one from its right argument.

$$\underline{\mathsf{dSum1}}\ \frac{t \xrightarrow{\underline{\alpha}} t'}{t \oplus u \xrightarrow{\underline{\alpha}} t'} \qquad \mathsf{dSum1}\ \frac{t \xrightarrow{m,\alpha} t'}{t \oplus u \xrightarrow{m\cdot L,\alpha} t'} \qquad \underline{\mathsf{dSum2}}\ \frac{u \xrightarrow{\underline{\alpha}} u'}{t \oplus u \xrightarrow{\underline{\alpha}} u'} \qquad \mathsf{dSum2}\ \frac{u \xrightarrow{n,\alpha} u'}{t \oplus u \xrightarrow{n\cdot R,\alpha} u'}$$

It can easily be checked that $\simeq^{\mathsf{l}}$ is also compositional with respect to $\oplus$.

TABLE 4.4
Axioms E (continued)

(D1)	$(t \oplus t') + (u \oplus u') = ((t \oplus t') + u') \oplus ((u \oplus u') + t')$	$(\vdash_I t \sqsubseteq_{\mathrm{i}} t' , \vdash_I u \sqsubseteq_{\mathrm{i}} u')$
(D2)	$(t \oplus u) + \underline{\alpha}.v = (t + \underline{\alpha}.v) \oplus (u + \underline{\alpha}.v)$	
(Ic1)	$t \oplus \underline{\alpha}.u = t + \underline{\alpha}.u$	$(\natural t)$
(Ic2)	$(\underline{\alpha}.t + u) = (\underline{\alpha}.t + u) \oplus \underline{\alpha}.t$	
(S1)	$(t + \underline{\alpha}.u) \oplus (t' + \underline{\alpha}.u') = (t + \underline{\alpha}.u + \underline{\alpha}.u') \oplus (t' + \underline{\alpha}.u')$	
(S2)	$(t + \alpha.v) \oplus (u + \alpha.v) = (t + \alpha.v) \oplus u$	$(\vdash_I t \sqsubseteq_{\mathrm{i}} u)$
(S3)	$t \oplus u = t + u$	$(\vdash_I t =_{\mathrm{i}} u)$

Now, we turn to the axiom system for distributed prioritized strong bisimulation. We write $\vdash_E t = u$ if term t can be rewritten to u using the axioms in Tables 4.3 and 4.4 as well as Axioms (A1)–(A4), Axioms (Res1)–(Res4), Axioms (Rel1)–(Rel3), and Axiom (P) from Table 3.3. Axioms (Ic1), (D1), (S2), and (S3) involve side conditions. Regarding Axiom (Ic1), we introduce the unary predicate $\natural$ over processes (of the form $\sum_{j \in J} \gamma_j.t_j$ for some nonempty index set J) together with the following proof rules: (i) $\natural \underline{\alpha}.t$ and (ii) $\natural t$ and $\natural u$ implies $\natural(t + u)$. Intuitively, $\natural(\sum_{j \in J} \gamma_j.t_j)$ if and only if $\gamma_j \in \underline{A}$ for all $j \in J$. The relation $\sqsubseteq_{\mathrm{i}}$ is defined as in Section 3.5 (see Table 3.8). The axioms in Table 4.3 are basically those given in Table 3.3 and augmented with the corresponding axioms for the distributed summation operator. Moreover, the Expansion Axiom has been adapted for our algebra (cf. Axiom (E) where $\sum$ is the indexed version of $+$, and $\bigoplus$ is the indexed version of $\oplus$). Note that parallelism in $\mathsf{CCS}^{\mathsf{sl}}$ cannot be resolved in nondeterminism by using the operator $+$ only, since priorities on different sides of $\mid$ are incomparable, but on different sides of $+$ they are comparable. The introduction of the operator $\oplus$ solves this problem. The axioms in Table 4.4 show how we may "restructure" locations. They deal with the *distributivity* of the summation operators (Axioms (D1) and (D2)), the *interchangeability* of the summation operators (Axioms (Ic1) and (Ic2)), and the *saturation* of locations (Axioms (S1), (S2), and (S3)), respectively. The proof of the next theorem can be found in [27].

THEOREM 4.5. *Let t and u be finite processes. Then $\vdash_E t = u$ if and only if $t \simeq^{\mathrm{l}} u$.*

4.3.3. Operational Characterization. The following definition introduces an equivalence $\simeq_*$ which characterizes $\simeq^{\mathrm{l}}$ as standard strong bisimulation [50]. It uses the notation $P \xrightarrow[L]{\alpha} P'$ for $P, P' \in \mathcal{P}$, $\alpha \in A$, and $L \subseteq \underline{A} \setminus \{\underline{\tau}\}$ whenever $\exists m \in \mathcal{L}oc.\, P \xrightarrow{m,\alpha} P'$ and $\underline{\mathit{IL}}_{[m]}(P) \subseteq L$. Note that these enriched transitions take local pre-emption potential into account, thereby avoiding the explicit annotation of transitions with locations.

DEFINITION 4.6. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is an* alternative distributed prioritized strong bisimulation *if for every $\langle P, Q \rangle \in \mathcal{R}$, $\alpha \in A$, $\underline{\alpha} \in \underline{A}$, and $L \subseteq \underline{A} \setminus \{\underline{\tau}\}$ the following conditions hold.*

1. *$P \xrightarrow{\underline{\alpha}} P'$ implies $\exists Q'.\, Q \xrightarrow{\underline{\alpha}} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.*
2. *$P \xrightarrow[L]{\alpha} P'$ implies $\exists Q'.\, Q \xrightarrow[L]{\alpha} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.*

We write $P \simeq_* Q$ if $\langle P, Q \rangle \in \mathcal{R}$ for some alternative distributed prioritized strong bisimulation $\mathcal{R}$.

Similar as in Section 3.3.3 we obtain an operational characterization of our behavioral relation.

THEOREM 4.7 (Operational Characterization). $\simeq^{\mathrm{l}} = \simeq_*$.

4.4. Semantic Theory Based on Weak Bisimulation. As for $\mathsf{CCS}^{\mathsf{sg}}$, we develop a coarser behavioral bisimulation-based congruence by abstracting from internal actions. We start off with the definition of a naive distributed prioritized weak bisimulation, which is an adaptation of observational equivalence [52].

DEFINITION 4.8 (Naive Distributed Prioritized Weak Transition Relation).

(i) $\hat{\gamma} =_{df} \epsilon$, *if* $\gamma \in \{\underline{\tau}, \tau\}$, *and* $\hat{\gamma} =_{df} \gamma$, *otherwise* *(ii)* $\stackrel{\epsilon}{\Longrightarrow}_{\times} =_{df} (\stackrel{\underline{\tau}}{\longrightarrow} \cup \bigcup\{ \stackrel{m,\tau}{\longrightarrow} \mid m \in \mathcal{L}oc\})^*$

(iii) $\stackrel{\underline{\alpha}}{\Longrightarrow}_{\times} =_{df} \stackrel{\epsilon}{\Longrightarrow}_{\times} \circ \stackrel{\underline{\alpha}}{\longrightarrow} \circ \stackrel{\epsilon}{\Longrightarrow}_{\times}$ *(iv)* $\stackrel{m,\alpha}{\Longrightarrow}_{\times} =_{df} \stackrel{\epsilon}{\Longrightarrow}_{\times} \circ \stackrel{m,\alpha}{\longrightarrow} \circ \stackrel{\epsilon}{\Longrightarrow}_{\times}$

In the following we write $P \stackrel{\alpha}{\Longrightarrow}_{\times} P'$ for $\exists m \in \mathcal{L}oc.\, P \stackrel{m,\alpha}{\Longrightarrow}_{\times} P'$.

DEFINITION 4.9 (Naive Distributed Prioritized Weak Bisimulation). *A symmetric relation* $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ *is a* naive distributed prioritized weak bisimulation *if for every* $\langle P, Q\rangle \in \mathcal{R}$ *and* $\gamma \in \mathcal{A}$ *the following condition holds.*

$$P \stackrel{\gamma}{\longrightarrow} P' \text{ implies } \exists Q'.\, Q \stackrel{\hat{\gamma}}{\Longrightarrow}_{\times} Q' \text{ and } \langle P', Q'\rangle \in \mathcal{R}\ .$$

We write $P \approx_{\times} Q$ *if* $\langle P, Q\rangle \in \mathcal{R}$ *for some naive distributed prioritized weak bisimulation* $\mathcal{R}$.

It is fairly easy to see that $\approx_{\times}$ is not a congruence for $\mathsf{CCS}^{\mathsf{sl}}$. One compositionality defect arises with respect to parallel composition and is similar to the one mentioned for naive distributed prioritized strong bisimulation. Another defect, which is carried over from CCS, is concerned with the summation operators.

4.4.1. Distributed Prioritized Weak Bisimulation. We devote the rest of this section to characterizing the largest congruence contained in the naive distributed prioritized weak bisimulation. To do so, we first re-define the weak transition relation.

DEFINITION 4.10 (Distributed Prioritized Weak Transition Relation). *For* $L, M \subseteq \underline{\mathcal{A}} \setminus \{\underline{\tau}\}$ *we define the following notations.*

(i) $\hat{\underline{\tau}} =_{df} \underline{\epsilon}$, $\hat{\underline{a}} =_{df} \underline{a}$, $\hat{\tau} =_{df} \epsilon$, $\hat{a} =_{df} a$ *(ii)* $P \stackrel{m,\alpha}{\underset{L}{\longrightarrow}} P'$ *if* $P \stackrel{m,\alpha}{\longrightarrow} P'$ *and* $\underline{\Pi}_{[m]}(P) \subseteq L$

(iii) $\stackrel{\epsilon}{\Longrightarrow} =_{df} (\stackrel{\underline{\tau}}{\longrightarrow} \cup \bigcup\{ \stackrel{m,\tau}{\underset{\emptyset}{\longrightarrow}} \mid m \in \mathcal{L}oc\})^*$ *(iv)* $\stackrel{\underline{\alpha}}{\Longrightarrow} =_{df} \stackrel{\epsilon}{\Longrightarrow} \circ \stackrel{\underline{\alpha}}{\longrightarrow} \circ \stackrel{\epsilon}{\Longrightarrow}$

(v) $\stackrel{\epsilon}{\underset{L}{\Longrightarrow}} =_{df} (\stackrel{\underline{\tau}}{\longrightarrow} \cup \bigcup\{ \stackrel{m,\tau}{\underset{L}{\longrightarrow}} \mid m \in \mathcal{L}oc\})^*$ *(vi)* $P \stackrel{m,\alpha}{\underset{L,M}{\Longrightarrow}} P'$ *if* $\exists P''.\, P \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} P'' \stackrel{m,\alpha}{\underset{L}{\longrightarrow}} \stackrel{\epsilon}{\Longrightarrow} P'$ *and* $\underline{\Pi}(P'') \subseteq M$.

Intuitively, these definitions are designed to reflect constraints that a process environment must satisfy in order for the given transition to be enabled. Thus, $P \stackrel{m,\alpha}{\underset{L}{\longrightarrow}} P'$ means that P can engage in action α at location m to P' *provided that* the environment does not offer a prioritized communication involving actions in L. If the environment were to offer such a communication, the result would be a $\underline{\tau}$ at a comparable location to m in P, which would pre-empt the α. In a similar vein, $P \stackrel{\epsilon}{\Longrightarrow} P'$ holds if P can evolve to P' via a nonpre-emptable sequence of internal transitions, regardless of the environment's behavior. These internal transitions should therefore involve either $\underline{\tau}$, which can never be pre-empted, or τ, in which case no prioritized actions should be enabled at the same location. Likewise, $P \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} P'$ means that, so long as the environment does not offer to synchronize with P using the prioritized actions in L, the process P may engage in a sequence of internal computation steps and become P'. Finally, the M-parameter in $\stackrel{m,\alpha}{\underset{L,M}{\Longrightarrow}}$ provides a measure of the pre-emptive impact that a process can have on its environment. From the definition, $P \stackrel{m,\alpha}{\underset{L,M}{\Longrightarrow}} P'$ is true if P can engage in some internal computation followed by α, so long as the environment refrains from synchronizations in L, and then some nonpre-emptable internal computation to arrive at P'. In addition, the state at which α is enabled should only offer prioritized communications in M. Note that the definition of $P \stackrel{\epsilon}{\underset{L}{\Longrightarrow}} P'$ is in accordance with our intuition that internal actions, and therefore their locations, are unobservable. Moreover, an environment of P is not influenced by internal actions performed by P since priorities arising from different sides of the

parallel composition operator are incomparable. Therefore, the parameter M is unnecessary in the definition of the relation $\stackrel{\epsilon}{\Longrightarrow}_L$. Finally, for notational convenience $\stackrel{m,\epsilon}{\Longrightarrow}_{L,M}$ is interpreted as $\stackrel{\epsilon}{\Longrightarrow}_L$.

DEFINITION 4.11 (Distributed Prioritized Weak Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* distributed prioritized weak bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$, $\alpha \in A$, $\underline{\alpha} \in \underline{A}$, and $m \in \mathcal{L}oc$ the following conditions hold.*

1. $\exists Q', Q''.\, Q \stackrel{\epsilon}{\Longrightarrow} Q'' \stackrel{\epsilon}{\Longrightarrow} Q'$, $\underline{\Pi}(Q'') \subseteq \underline{\Pi}(P)$, *and* $\langle P, Q'\rangle \in \mathcal{R}$.
2. $P \stackrel{\underline{\alpha}}{\longrightarrow} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow} Q'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \stackrel{m,\underline{\alpha}}{\longrightarrow} P'$ *implies* $\exists Q', n.\, Q \stackrel{n,\hat{\underline{\alpha}}}{\Longrightarrow}_{L,M} Q'$, $L = \underline{\Pi}_{[m]}(P)$, $M = \underline{\Pi}(P)$, *and* $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \underline{\approx} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some distributed prioritized weak bisimulation $\mathcal{R}$.

Condition (1) of Definition 4.11 guarantees that distributed prioritized weak bisimulation is compositional with respect to parallel composition. Its necessity is best illustrated by the following example. The processes $P \stackrel{\text{def}}{=} \underline{\tau}.\underline{a}.\mathbf{0}$ and $Q \stackrel{\text{def}}{=} \underline{a}.\mathbf{0}$ would be considered equivalent if Condition (1) were absent. However, the context $C[X] \stackrel{\text{def}}{=} X\,|(\overline{\underline{a}}.\mathbf{0} + b.\mathbf{0})$ distinguishes them. The following proposition is the $\mathsf{CCS}^{\mathsf{s|}}$ equivalent of Proposition 3.9.

PROPOSITION 4.12. *The equivalence relation $\underline{\approx}$ is a congruence with respect to prefixing, parallel composition, relabeling, and restriction. Moreover, $\underline{\approx}$ is characterized as the* largest congruence *contained in $\approx_\times$, in the sub-algebra of $\mathsf{CCS}^{\mathsf{s|}}$ induced by these operators and recursion.*

4.4.2. Distributed Prioritized Observational Congruence. Analogue to Section 3, the summation fix presented in [52] is not sufficient in order to achieve a congruence relation.

DEFINITION 4.13. *We define $P \underline{\approx}^{\mathrm{l}} Q$ if for all $\alpha \in A$, $\underline{\alpha} \in \underline{A}$, and $m \in \mathcal{L}oc$ the following conditions and their symmetric counterparts hold.*

1. $\underline{\mathcal{I}}(P) \supseteq \underline{\mathcal{I}}(Q)$
2. $P \stackrel{\underline{\alpha}}{\longrightarrow} P'$ *implies* $\exists Q'.\, Q \stackrel{\underline{\alpha}}{\Longrightarrow} Q'$ *and* $P' \underline{\approx} Q'$.
3. $P \stackrel{m,\underline{\alpha}}{\longrightarrow} P'$ *implies* $\exists Q', n.\, Q \stackrel{n,\underline{\alpha}}{\Longrightarrow}_{L,M} Q'$, $L = \underline{\Pi}_{[m]}(P)$, $M = \underline{\Pi}(P)$, *and* $P' \underline{\approx} Q'$.

The following theorem can be proved by following the technique already presented in Section 3.3.2 (cf. [50]).

THEOREM 4.14. *$\underline{\approx}^{\mathrm{l}}$ is the largest congruence contained in $\approx_\times$.*

4.4.3. Operational Characterization. We now characterize distributed prioritized weak bisimulation as standard bisimulation over an appropriately defined transition relation. To begin with, we introduce a family of relations $\stackrel{}{\Longrightarrow}_M$ on processes, where $M \subseteq \underline{A} \setminus \{\underline{\tau}\}$, by defining $P \stackrel{}{\Longrightarrow}_M P'$ if $\exists P''.\, P \stackrel{\epsilon}{\Longrightarrow} P'' \stackrel{\epsilon}{\Longrightarrow} P'$ and $\underline{\Pi}(P'') \subseteq M$. Moreover, we write $P \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow}_{L,M} P'$ whenever there exists some $m \in \mathcal{L}oc$ such that $P \stackrel{m,\hat{\underline{\alpha}}}{\Longrightarrow}_{L,M} P'$.

DEFINITION 4.15. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is an* alternative distributed prioritized weak bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$, $\alpha \in A$, $\underline{\alpha} \in \underline{A}$, and $L, M \subseteq \underline{A} \setminus \{\underline{\tau}\}$ the following conditions hold.*

1. $P \stackrel{}{\Longrightarrow}_M P'$ *implies* $\exists Q'.\, Q \stackrel{}{\Longrightarrow}_M Q'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.
2. $P \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow} Q'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow}_{L,M} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\underline{\alpha}}}{\Longrightarrow}_{L,M} Q'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \underline{\approx}_* Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some alternative distributed prioritized weak bisimulation $\mathcal{R}$.

THEOREM 4.16 (Operational Characterization). $\underline{\approx} = \underline{\approx}_*$.

The interested reader can find the proof of this theorem in [50].

4.5. Example. We now return to the direct-memory-access example system introduced in the beginning of Section 4. The $\mathsf{CCS}^{\mathsf{s|}}$ semantics of `Sys`, which corresponds to our intuition regarding distributed systems, is given in Figure 4.2 where we abstract away the locations.

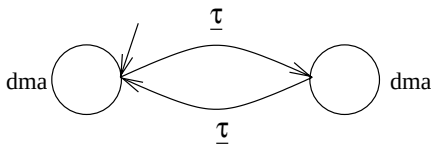

FIG. 4.2. *Semantics of the dma-system*

As stated before, the application uses the two memory cells alternately. Thus, the DMA is expected to be allowed to access the free memory bench. Accordingly, the specification of the system can be formalized by $\mathtt{Spec} \stackrel{\text{def}}{=} \mathtt{dma}.\mathtt{Spec}$. It is easy to see that the symmetric closure of

$$\{\langle \mathtt{Spec}, \mathtt{Sys}\rangle, \langle \mathtt{Spec}, (\overline{\mathtt{fetch2}}.\mathtt{Appl} \,|\, \mathtt{Bench1} \,|\, \mathtt{Bench2}) \setminus \{\underline{\mathtt{fetch1}}, \underline{\mathtt{fetch2}}\}\rangle\}$$

is a distributed prioritized weak bisimulation. Therefore, $\mathtt{Spec} \underline{\approx} \mathtt{Sys}$ as expected, i.e., the system `Sys` meets its specification `Spec`.

4.6. Discussion on the Removal of Some Restrictive Design Decisions. Up to now we have restricted the number of priority levels in $\mathsf{CCS}^{\mathsf{s|}}$ to two and communication to complementary actions having the same priority. In this section we study the implications of the removal of these restrictions leading to a new version of $\mathsf{CCS}^{\mathsf{s|}}$, called $\mathsf{CCS}^{\mathsf{s|}}_{\mathsf{ml}}$ ($\mathsf{CCS}^{\mathsf{s|}}$ with a multi-level priority-scheme), that is formally defined in the next section.

Allowing communication between unprioritized actions and complementary prioritized actions raises the question of whether the resulting internal action should be τ or $\underline{\tau}$. When dealing with local pre-emption, this decision has no important consequences for sequential communicating processes, i.e., those in *standard concurrent form* [52]; however, it is of obvious importance for processes like $(\underline{a}.\mathbf{0} \,|\, \overline{a}.\mathbf{0}) + b.\mathbf{0}$ in which one has to decide if the b-transition is enabled. One reasonable view is that a communication should be pre-empted whenever one communication partner is pre-empted, i.e., cannot engage in a communication. This implies that the *minimal* priority of the complementary actions ought to be assigned to the internal action. To reflect this in the operational semantics, one could replace Rules (Com1), (Com2), and (Com3) for parallel composition by the ones presented in Table 4.5 plus their symmetric versions. The side conditions involve sets $\mathcal{I}\!\!I(P)$ that include all unprioritized visible actions in which P can initially engage.

It turns out that the largest congruence results concerning distributed prioritized strong bisimulation and distributed prioritized observational congruence can be carried over to the new calculus; however, the new semantics has algebraic shortcomings, since parallel composition is *not associative*, as illustrated by the following example. Consider the process $(b.\mathbf{0} + \underline{a}.\mathbf{0}) \,|\, (\overline{a}.\mathbf{0} + \underline{c}.\mathbf{0}) \,|\, \overline{\underline{c}}.\mathbf{0}$. When computing the semantics in a left-associative manner, the initial b-transition is pre-empted according to Rule (Com1) since $\underline{a}$ may potentially communicate with $\overline{a}$. However, when first composing the second and third parallel components, the $\overline{a}$-transition is pre-empted, and consequently the b-transition is enabled by Rule (Com1). The reason for this problem is that transitions are pre-empted because the considered process can *potentially* engage in a higher prioritized communication from a comparable location. However, this potential communication cannot take place if the communication partner is itself pre-empted. The same problem also arises when extending $\mathsf{CCS}^{\mathsf{s|}}$ to multiple priority levels, even if communication is only allowed on complementary actions

TABLE 4.5
Modified operational rules

$$\textsf{Com1}\quad \frac{P \xrightarrow{m,\alpha} P'}{P|Q \xrightarrow{m\cdot L,\alpha} P'|Q}\ \underline{\!\!\Pi}_{[m]}(P) \cap (\overline{\underline{\!\!\Pi}(Q)} \cup \overline{\Pi(Q)}) = \emptyset$$

$$\textsf{Com3a}\quad \frac{P \xrightarrow{m,a} P' \quad Q \xrightarrow{n,\overline{a}} Q'}{P|Q \xrightarrow{\langle m\cdot L, n\cdot R\rangle,\tau} P'|Q'}\ \begin{array}{l} \underline{\!\!\Pi}_{[m]}(P) \cap (\overline{\underline{\!\!\Pi}(Q)} \cup \overline{\Pi(Q)}) = \emptyset \wedge \\ \underline{\!\!\Pi}_{[n]}(Q) \cap (\overline{\underline{\!\!\Pi}(P)} \cup \overline{\Pi(P)}) = \emptyset \end{array}$$

$$\textsf{Com3b}\quad \frac{P \xrightarrow{m,a} P' \quad Q \xrightarrow{n,\overline{a}} Q'}{P|Q \xrightarrow{\langle m\cdot L, n\cdot R\rangle,\tau} P'|Q'}\ \underline{\!\!\Pi}_{[n]}(Q) \cap (\overline{\underline{\!\!\Pi}(P)} \cup \overline{\Pi(P)}) = \emptyset$$

of the same priority as can be observed by using an adaptation of the previous example: $(b{:}2.\mathbf{0} + a{:}1.\mathbf{0}) \,|\, (\overline{a}{:}\, 1.\mathbf{0} + c{:}0.\mathbf{0}) \,|\, \overline{c}{:}0.\mathbf{0}$.

One can imagine two approaches to fixing the problems with the first (and second) alteration to the theory. One is to change the operational semantics; in particular, the side conditions could be weakened so that an unprioritized transition is only pre-empted when a prioritized action from a comparable location can *actually* engage in a communication. This approach has not been investigated in the literature, yet. The second solution follows an approach developed in [23] for a different setting and involves the use of a syntax restriction on processes prohibiting output actions, i.e., actions in $\overline{\Lambda}$, from occurring as initial actions of processes that are in the scope of $+$. Hence, all potential communication partners are also actual ones, and the standard side conditions for parallel composition are sufficient to encode the desired notion of pre-emption. It is important to mention that the proposed syntax restriction still allows one to specify many practically relevant examples within the calculus. Indeed, a similar restriction may be found in the programming language `occam` [41].

4.7. Extension to Multi-level Priority-schemes. For $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$, we allow a multi-level priority-scheme and communication between complementary actions with potentially different priorities. As seen in the previous section, both of these relaxations yield a semantics for which parallel composition is not associative. However, we have also argued that this problem vanishes if the syntax is restricted such that output actions never get pre-empted. We adapt the syntax restriction proposed by Camilleri and Winskel [23], stating that initial actions in the scope of a comparable summation operator are input actions. Therefore, input and output actions are explicitly distinguished in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$, where the internal action τ is also treated as input action. In the following, we let $a, b, \ldots$ range over the set Λ of input ports and $\overline{a}, \overline{b}, \ldots$ over the set $\overline{\Lambda}$ of output ports. Moreover, we let γ stand for the silent action τ or an input action and let α range over $\mathcal{A} =_{\mathrm{df}} \Lambda \cup \overline{\Lambda} \cup \{\tau\}$. Since the priority values of output actions need never be compared with other priority values in the restricted syntax, there are no priority values associated with output actions. The syntax of $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ is formally defined by the following BNF for P.

$$\begin{array}{lcl} I & ::= & \mathbf{0} \mid x \mid \gamma{:}k.I \mid I + I \mid I \oplus I \mid I \,|\, I \mid I[f] \mid I \setminus L \mid \mu x.I \\ P & ::= & \mathbf{0} \mid x \mid \alpha{:}k.P \mid I + I \mid P \oplus P \mid P \,|\, P \mid P[f] \mid P \setminus L \mid \mu x.P \end{array}$$

Here, f is an *injective*, finite relabeling, $L \subseteq \Lambda \cup \overline{\Lambda}$ is a restriction set, and x is a variable taken from a countable domain $\mathcal{V}$. A relabeling satisfies the properties $f(\Lambda) \subseteq \Lambda$, $f(\overline{\Lambda}) \subseteq \overline{\Lambda}$, $f(\tau) = \tau$, and $f(\overline{a}) = \overline{f(a)}$.

Thus, additionally to the requirements of a finite relabeling in CCS, relabelings in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ may only map input ports to input ports and output ports to output ports. Since actions attached with different priority values do not represent different ports here, relabelings and restriction sets do not deal with priority values. Thus, the priority value of a relabeled transition remains the same, i.e., there is no implicit mechanism for prioritization or deprioritization (cf. Section 3.5). In the sequel, we write $\mathcal{P}^{\mathsf{sl}}_{\mathsf{ml}}$ for the set of $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ processes.

TABLE 4.6
Initial output action sets for $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$

$\overline{\mathit{I}}(\mu x.P)$	$=$	$\overline{\mathit{I}}(P[\mu x.P/x])$	$\overline{\mathit{I}}(\overline{a}.P)$	$=$	$\{a\}$
$\overline{\mathit{I}}(P \mid Q)$	$=$	$\overline{\mathit{I}}(P) \cup \overline{\mathit{I}}(Q)$	$\overline{\mathit{I}}(P \oplus Q)$	$=$	$\overline{\mathit{I}}(P) \cup \overline{\mathit{I}}(Q)$
$\overline{\mathit{I}}(P[f])$	$=$	$\{f(a) \mid a \in \overline{\mathit{I}}(P)\}$	$\overline{\mathit{I}}(P \setminus L)$	$=$	$\overline{\mathit{I}}(P) \setminus (L \cup \overline{L})$

The semantics of $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ processes are again labeled transition systems whose transition relations are specified by operational rules. Since output transitions cannot get pre-empted they do also not need an associated priority value, and output transitions do not need to take account of locations. We first present two auxiliary sets used when presenting the operational rules, namely (i) initial output action sets $\overline{\mathit{I}}(P)$ of a process P and (ii) initial input action sets $\mathrm{I}^k_m(P)$ of P with respect to a priority value k and a location m, which are defined to be the smallest sets satisfying the equations presented in Tables 4.6 and 4.7, respectively. For technical convenience we remove the complement of output actions in the definition of $\overline{\mathit{I}}(\cdot)$, and we use the following abbreviations: (i) $\mathrm{I}^{<k}_M(P) =_{\mathrm{df}} \bigcup\{\mathrm{I}^l_m(P) \mid m \in M,\ l < k\}$, (ii) $\mathrm{II}^{<k}_M(P) =_{\mathrm{df}} \mathrm{I}^{<k}_M(P) \setminus \{\tau\}$, (iii) $\mathrm{I}(P) =_{\mathrm{df}} \bigcup\{\mathrm{I}^l_m(P) \mid m \in \mathcal{L}oc,\ l \in \mathcal{N}\}$, and (iv) $\mathrm{II}(P) =_{\mathrm{df}} \mathrm{I}(P) \setminus \{\tau\}$.

TABLE 4.7
Initial input action sets for $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$

$\mathrm{I}^k_m(\mu x.P)$	$=$	$\mathrm{I}^k_m(P[\mu x.P/x])$	$\mathrm{I}^k_\bullet(\gamma{:}l.P)$	$=$	$\{\gamma \mid k = l\}$
$\mathrm{I}^k_{m\cdot l}(P + Q)$	$=$	$\mathrm{I}^k_m(P)$	$\mathrm{I}^k_{m\cdot L}(P \oplus Q)$	$=$	$\mathrm{I}^k_m(P)$
$\mathrm{I}^k_{n\cdot r}(P + Q)$	$=$	$\mathrm{I}^k_n(Q)$	$\mathrm{I}^k_{n\cdot R}(P \oplus Q)$	$=$	$\mathrm{I}^k_n(Q)$
$\mathrm{I}^k_m(P[f])$	$=$	$\{f(\gamma) \mid \gamma \in \mathrm{I}^k_m(P)\}$	$\mathrm{I}^k_{m\cdot L}(P \mid Q)$	$=$	$\mathrm{I}^k_m(P) \cup \{\tau \mid \mathrm{I}^k_m(P) \cap \overline{\mathit{I}}(Q) \neq \emptyset\}$
$\mathrm{I}^k_m(P \setminus L)$	$=$	$\mathrm{I}^k_m(P) \setminus (L \cup \overline{L})$	$\mathrm{I}^k_{n\cdot R}(P \mid Q)$	$=$	$\mathrm{I}^k_n(Q) \cup \{\tau \mid \mathrm{I}^k_n(Q) \cap \overline{\mathit{I}}(P) \neq \emptyset\}$

The operational rules for $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ semantics are formally stated in Table 4.8 for output transitions and in Table 4.9 for input transitions. As expected, the rules for output transitions coincide with the ones for plain CCS [52] whereas the rules for input transitions take local pre-emption into account, thereby using location and priority value information in their side conditions. It is worth having a closer look at the side conditions of Rules (Sum1) and (Sum2) which differ in principle from the corresponding ones of $\mathsf{CCS}^{\mathsf{sl}}$. They guarantee that an initial $\gamma{:}l$-transition of a process P is also pre-empted whenever there exists a higher prioritized initial $\gamma{:}k$-transition of P, i.e., if $k < l$. This additional kind of pre-emption reflects that output transitions can communicate with a complementary input transition regardless of its priority value, i.e., if more than one communication partner offering the matching input transition is available from comparable locations, the one attached with the highest priority is taken. This kind of pre-emption requires relabelings

TABLE 4.8
Operational semantics for CCS^{sl}_{ml} wrt. output transitions

$$\overline{\mathsf{Act}}\quad \frac{-}{\overline{a}.P \xrightarrow{\overline{a}} P} \qquad \overline{\mathsf{iSum1}}\quad \frac{P \xrightarrow{\overline{a}} P'}{P \oplus Q \xrightarrow{\overline{a}} P'} \qquad \overline{\mathsf{Com1}}\quad \frac{P \xrightarrow{\overline{a}} P'}{P \mid Q \xrightarrow{\overline{a}} P' \mid Q}$$

$$\overline{\mathsf{Rel}}\quad \frac{P \xrightarrow{\overline{a}} P'}{P[f] \xrightarrow{f(\overline{a})} P'[f]} \qquad \overline{\mathsf{iSum2}}\quad \frac{Q \xrightarrow{\overline{a}} Q'}{P \oplus Q \xrightarrow{\overline{a}} Q'} \qquad \overline{\mathsf{Com2}}\quad \frac{Q \xrightarrow{\overline{a}} Q'}{P \mid Q \xrightarrow{\overline{a}} P \mid Q'}$$

$$\overline{\mathsf{Rec}}\quad \frac{P[\mu x.P/x] \xrightarrow{\overline{a}} P'}{\mu x.P \xrightarrow{\overline{a}} P'} \qquad \overline{\mathsf{Res}}\quad \frac{P \xrightarrow{\overline{a}} P'}{P \setminus L \xrightarrow{\overline{a}} P' \setminus L}\ \overline{a} \notin L \cup \overline{L}$$

TABLE 4.9
Operational semantics for CCS^{sl}_{ml} wrt. input transitions

$$\mathsf{Act}\quad \frac{-}{\gamma{:}k.P \xrightarrow{\bullet,\gamma:k} P} \qquad \mathsf{Sum1}\quad \frac{P \xrightarrow{m,\gamma:k} P'}{P + Q \xrightarrow{m\cdot l,\gamma:k} P'}\ \tau,\gamma \notin \mathrm{I}^{<k}(Q)$$

$$\mathsf{iSum1}\quad \frac{P \xrightarrow{m,\gamma:k} P'}{P \oplus Q \xrightarrow{m\cdot L,\gamma:k} P'} \qquad \mathsf{Sum2}\quad \frac{Q \xrightarrow{n,\gamma:k} Q'}{P + Q \xrightarrow{n\cdot r,\gamma:k} Q'}\ \tau,\gamma \notin \mathrm{I}^{<k}(P)$$

$$\mathsf{iSum2}\quad \frac{Q \xrightarrow{n,\gamma:k} Q'}{P \oplus Q \xrightarrow{n\cdot R,\gamma:k} Q'} \qquad \mathsf{Com1}\quad \frac{P \xrightarrow{m,\gamma:k} P'}{P \mid Q \xrightarrow{m\cdot L,\gamma:k} P' \mid Q}\ \mathrm{I\!I}^{<k}_{[m]}(P) \cap \overline{\mathit{I\!I}}(Q) = \emptyset$$

$$\mathsf{Rel}\quad \frac{P \xrightarrow{m,\gamma:k} P'}{P[f] \xrightarrow{m,f(\gamma):k} P'[f]} \qquad \mathsf{Com2}\quad \frac{Q \xrightarrow{n,\gamma:k} Q'}{P \mid Q \xrightarrow{n\cdot R,\gamma:k} P \mid Q'}\ \mathrm{I\!I}^{<k}_{[n]}(Q) \cap \overline{\mathit{I\!I}}(P) = \emptyset$$

$$\mathsf{Rec}\quad \frac{P[\mu x.P/x] \xrightarrow{m,\gamma:k} P'}{\mu x.P \xrightarrow{m,\gamma:k} P'} \qquad \mathsf{Com3}\quad \frac{P \xrightarrow{m,a:k} P' \quad Q \xrightarrow{\overline{a}} Q'}{P \mid Q \xrightarrow{m\cdot L,\tau:k} P' \mid Q'}\ \mathrm{I\!I}^{<k}_{[m]}(P) \cap \overline{\mathit{I\!I}}(Q) = \emptyset$$

$$\mathsf{Res}\quad \frac{P \xrightarrow{m,\gamma:k} P'}{P \setminus L \xrightarrow{m,\gamma:k} P' \setminus L}\ \gamma \notin L \cup \overline{L} \qquad \mathsf{Com4}\quad \frac{P \xrightarrow{\overline{a}} P' \quad Q \xrightarrow{n,a:k} Q'}{P \mid Q \xrightarrow{n\cdot R,\tau:k} P' \mid Q'}\ \mathrm{I\!I}^{<k}_{[n]}(Q) \cap \overline{\mathit{I\!I}}(P) = \emptyset$$

to be restricted to injective ones as is pointed out in [23].

The behavioral relations defined for $\mathsf{CCS}^{\mathsf{sl}}$ can be adapted to $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ in a straightforward fashion, as we demonstrate by the notion of distributed prioritized strong bisimulation.

DEFINITION 4.17. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* distributed prioritized strong bisimulation for $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ *if for every $\langle P, Q\rangle \in \mathcal{R}$, $\overline{a} \in \overline{\Lambda}$, $\gamma \in \Lambda \cup \{\tau\}$, $k \in \mathcal{N}$, and $m \in \mathcal{L}oc$, the following conditions hold.*

1. *$P \xrightarrow{\overline{a}} P'$ implies $\exists Q'.\, Q \xrightarrow{\overline{a}} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$, and*
2. *$P \xrightarrow{m,\gamma:k} P'$ implies $\exists Q', l, n.\, Q \xrightarrow{n,\gamma:l} Q'$, $\mathrm{I\!I}^{<l}_{[n]}(Q) \subseteq \mathrm{I\!I}^{<k}_{[m]}(P)$, and $\langle P', Q'\rangle \in \mathcal{R}$.*

We write $P \simeq_{ml} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for a distributed prioritized strong bisimulation $\mathcal{R}$ for CCS^{sl}_{ml}.

PROPOSITION 4.18. *The relation $\simeq_{ml}$ is compositional with respect to all operators except summation.*

The proof can be done by using standard techniques [52] and, therefore, is omitted here. The reason for the compositionality lack with respect to summation is illustrated by the following example: $a{:}0.\mathbf{0} \simeq_{\mathsf{ml}} a{:}1.\mathbf{0}$ holds, but $a{:}0.\mathbf{0} + \tau{:}0.\mathbf{0} \not\simeq_{\mathsf{ml}} a{:}1.\mathbf{0} + \tau{:}0.\mathbf{0}$ since the former process can engage in a transition labeled by action a whereas the latter cannot. Although this defect can easily be repaired (note the analogy with weak bisimulation [52]) we do not elaborate on this further since it is not of importance here.

4.8. Camilleri and Winskel's Approach. Here, we briefly review Camilleri and Winskel's approach to priority [23], which we refer to as $\mathsf{CCS}^{\mathsf{cw}}$ (CCS with priority due to Camilleri and Winskel). In contrast to the approaches considered so far, this process algebra with priority does not assign priority values to actions. Instead, there exists a special summation operator $+\!\!\rangle$ in $\mathsf{CCS}^{\mathsf{cw}}$, called *prioritized choice*, which favors its left over its right argument. The syntax of $\mathsf{CCS}^{\mathsf{cw}}$ terms is given by the following BNF for P.

$$\begin{array}{lcl} I & ::= & \mathbf{0} \mid x \mid \gamma.I \mid I +\!\!\rangle I \mid \ I + I \ \mid \ I \,|\, I \ \mid I[f] \mid I \setminus L \mid \mu x.I \\ P & ::= & \mathbf{0} \mid x \mid \alpha.P \mid I +\!\!\rangle I \mid P + P \mid P \,|\, P \mid P[f] \mid P \setminus L \mid \mu x.P \end{array}$$

Here, the action γ, the injective, finite relabeling f, and the restriction set L satisfy the same restrictions as in the previous section. Again, closed and guarded terms determine the set $\mathcal{P}^{\mathsf{cw}}$ of $\mathsf{CCS}^{\mathsf{cw}}$ processes. Further, we introduce initial output and input action sets as displayed in Tables 4.10 and 4.11, respectively, and write $\mathbb{I}^{\mathsf{cw}}(P)$ for $\mathrm{I}^{\mathsf{cw}}(P) \setminus \{\tau\}$.

TABLE 4.10
Initial output action sets for $\mathsf{CCS}^{\mathsf{cw}}$

$\overline{\mathbb{I}}^{\mathsf{cw}}(\overline{a}.P)$	$=$	$\{a\}$	$\overline{\mathbb{I}}^{\mathsf{cw}}(\mu x.P)$	$=$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P[\mu x.P/x])$
$\overline{\mathbb{I}}^{\mathsf{cw}}(P \,\|\, Q)$	$=$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P) \cup \overline{\mathbb{I}}^{\mathsf{cw}}(Q)$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P + Q)$	$=$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P) \cup \overline{\mathbb{I}}^{\mathsf{cw}}(Q)$
$\overline{\mathbb{I}}^{\mathsf{cw}}(P[f])$	$=$	$\{f(a) \mid a \in \overline{\mathbb{I}}^{\mathsf{cw}}(P)\}$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P \setminus L)$	$=$	$\overline{\mathbb{I}}^{\mathsf{cw}}(P) \setminus (L \cup \overline{L})$

TABLE 4.11
Initial input action sets for $\mathsf{CCS}^{\mathsf{cw}}$

$\mathrm{I}^{\mathsf{cw}}(\gamma.P)$	$=$	$\{\gamma\}$	$\mathrm{I}^{\mathsf{cw}}(\mu x.P)$	$=$	$\mathrm{I}^{\mathsf{cw}}(P[\mu x.P/x])$
$\mathrm{I}^{\mathsf{cw}}(P +\!\!\rangle Q)$	$=$	$\mathrm{I}^{\mathsf{cw}}(P) \cup \mathrm{I}^{\mathsf{cw}}(Q)$	$\mathrm{I}^{\mathsf{cw}}(P + Q)$	$=$	$\mathrm{I}^{\mathsf{cw}}(P) \cup \mathrm{I}^{\mathsf{cw}}(Q)$
$\mathrm{I}^{\mathsf{cw}}(P[f])$	$=$	$\{f(\gamma) \mid \gamma \in \mathrm{I}^{\mathsf{cw}}(P)\}$	$\mathrm{I}^{\mathsf{cw}}(P \setminus L)$	$=$	$\mathrm{I}^{\mathsf{cw}}(P) \setminus (L \cup \overline{L})$

$$\mathrm{I}^{\mathsf{cw}}(P \,|\, Q) = \mathrm{I}^{\mathsf{cw}}(P) \cup \mathrm{I}^{\mathsf{cw}}(Q) \cup \{\tau \mid \mathrm{I}^{\mathsf{cw}}(P) \cap \overline{\mathbb{I}}^{\mathsf{cw}}(Q) \neq \emptyset\}$$

The semantics of a $\mathsf{CCS}^{\mathsf{cw}}$ process is given by a labeled transition system whose transition relation gives rise to transitions of the form $\vdash^{\mathsf{cw}}_M P \xrightarrow{\alpha} P'$, where $M \subseteq \Lambda$. Intuitively, process P can engage in an α-transition to P' whenever the environment does not offer communications on ports in M. Despite notational differences, this is the same underlying principle as for some transition relations defined in the previous sections which are also parameterized by initial action sets. Note that $\alpha \in \overline{\Lambda}$ implies $M = \emptyset$. The $\mathsf{CCS}^{\mathsf{cw}}$ transition relation is formally defined in Table 4.12, where $f(M)$ stands for $\{f(m) \mid m \in M\}$. Recall that the initial actions of P in $P +\!\!\rangle Q$ are given preference over the initial actions of Q. Also, in this approach a

prioritized τ, i.e., an internal action in which the left argument of $+\!\rangle$ can initially engage, has pre-emptive power over unprioritized actions, i.e., actions in which the right argument of $+\!\rangle$ can initially engage. Thus, the prioritized choice operator $+\!\rangle$ of [23] corresponds to the summation operator $+$ in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$. In [23] the operator $+$ stands for nondeterministic choice where priorities arising from the left and the right argument are incomparable. This operator is matched by the distributed summation operator $\oplus$ in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$. We further investigate the correspondence of these operators in the next section.

TABLE 4.12
Operational semantics for $\mathsf{CCS}^{\mathsf{cw}}$

Act	$\dfrac{-}{\vdash^{\mathsf{cw}}_{\emptyset}\ \alpha.P \xrightarrow{\alpha} P}$	Res	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{M\setminus(L\cup\overline{L})}\ P \setminus L \xrightarrow{\alpha} P' \setminus L}\ \alpha \notin L \cup \overline{L}$
Sum1	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{M}\ P +\!\rangle\, Q \xrightarrow{\alpha} P'}$	Sum2	$\dfrac{\vdash^{\mathsf{cw}}_{N}\ Q \xrightarrow{\alpha} Q'}{\vdash^{\mathsf{cw}}_{N\cup \mathrm{I}^{\mathsf{cw}}(P)}\ P +\!\rangle\, Q \xrightarrow{\alpha} Q'}\ \tau, \alpha \notin \mathrm{I}^{\mathsf{cw}}(P)$
iSum1	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{M}\ P + Q \xrightarrow{\alpha} P'}$	Com1	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{M}\ P \,|\, Q \xrightarrow{\alpha} P' \,|\, Q}\ M \cap \overline{\mathcal{L}}^{\mathsf{cw}}(Q) = \emptyset$
iSum2	$\dfrac{\vdash^{\mathsf{cw}}_{N}\ Q \xrightarrow{\alpha} Q'}{\vdash^{\mathsf{cw}}_{N}\ P + Q \xrightarrow{\alpha} Q'}$	Com2	$\dfrac{\vdash^{\mathsf{cw}}_{N}\ Q \xrightarrow{\alpha} Q'}{\vdash^{\mathsf{cw}}_{N}\ P \,|\, Q \xrightarrow{\alpha} P \,|\, Q'}\ N \cap \overline{\mathcal{L}}^{\mathsf{cw}}(P) = \emptyset$
Rel	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{f(M)}\ P[f] \xrightarrow{f(\alpha)} P'[f]}$	Com3	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P \xrightarrow{a} P'\ \vdash^{\mathsf{cw}}_{\emptyset}\ Q \xrightarrow{\overline{a}} Q'}{\vdash^{\mathsf{cw}}_{M}\ P \,|\, Q \xrightarrow{\tau} P' \,|\, Q'}\ M \cap \overline{\mathcal{L}}^{\mathsf{cw}}(Q) = \emptyset$
Rec	$\dfrac{\vdash^{\mathsf{cw}}_{M}\ P[\mu x.P/x] \xrightarrow{\alpha} P'}{\vdash^{\mathsf{cw}}_{M}\ \mu x.P \xrightarrow{\alpha} P'}$	Com4	$\dfrac{\vdash^{\mathsf{cw}}_{\emptyset}\ P \xrightarrow{\overline{a}} P'\ \vdash^{\mathsf{cw}}_{N}\ Q \xrightarrow{a} Q'}{\vdash^{\mathsf{cw}}_{N}\ P \,|\, Q \xrightarrow{\tau} P' \,|\, Q'}\ N \cap \overline{\mathcal{L}}^{\mathsf{cw}}(P) = \emptyset$

Camilleri and Winskel have also developed a bisimulation-based semantic theory for $\mathsf{CCS}^{\mathsf{cw}}$. Their notion of strong bisimulation for $\mathsf{CCS}^{\mathsf{cw}}$, as defined below, is shown to be a congruence [23].

DEFINITION 4.19. *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* distributed prioritized strong bisimulation for $\mathsf{CCS}^{\mathsf{cw}}$ *if for every $\langle P, Q\rangle \in \mathcal{R}$, $\alpha \in \mathcal{A}$, and $M \subseteq \Lambda$ the following condition holds:*

$$\vdash^{cw}_{M} P \xrightarrow{\alpha} P' \ \textit{ implies } \ \exists Q', N.\ \vdash^{cw}_{N} Q \xrightarrow{\alpha} Q',\ N \subseteq M,\ \textit{ and } \langle P', Q'\rangle \in \mathcal{R}\ .$$

We write $P \simeq_{cw} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some distributed prioritized strong bisimulation $\mathcal{R}$ for CCS^{cw}.

4.9. Relating Both Priority Approaches. In this section we show that $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ and $\mathsf{CCS}^{\mathsf{cw}}$ are closely related by providing an embedding of $\mathsf{CCS}^{\mathsf{cw}}$ in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$. For this purposes we define $\mathcal{N} =_{\mathrm{df}} \{0,1\}^*$ and the strict order $<$ on priority values to be the lexicographical order on $\mathcal{N}$, where 1 is less than 0.

We now introduce the *translation function* $\xi(\cdot) : \mathcal{P}^{\mathsf{cw}} \longrightarrow \mathcal{P}^{\mathsf{sl}}_{\mathsf{ml}}$ by defining $\xi(P) =_{\mathrm{df}} \xi^{\epsilon}(P)$, which maps $\mathsf{CCS}^{\mathsf{cw}}$ terms to $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ terms. The functions $\xi^{k}(P)$, for $k \in \mathcal{N}$, are inductively defined over the structure of $\mathsf{CCS}^{\mathsf{cw}}$ processes as shown in Table 4.13. We note that the translation function is not *surjective*, e.g.,

TABLE 4.13
Translation function

$\xi^k(\mathbf{0})$	$=_{\mathrm{df}}$	$\mathbf{0}$	$\xi^k(P+Q)$	$=_{\mathrm{df}}$	$\xi^k(P) \oplus \xi^k(Q)$	$\xi^k(P \setminus L)$	$=_{\mathrm{df}}$	$\xi^k(P) \setminus L$
$\xi^k(x)$	$=_{\mathrm{df}}$	x	$\xi^k(P \mathbin{+\!\!\rangle} Q)$	$=_{\mathrm{df}}$	$\xi^{k0}(P) + \xi^{k1}(Q)$	$\xi^k(P[f])$	$=_{\mathrm{df}}$	$\xi^k(P)[f]$
$\xi^k(\gamma.P)$	$=_{\mathrm{df}}$	$\gamma{:}k.\xi^\epsilon(P)$	$\xi^k(P \mid Q)$	$=_{\mathrm{df}}$	$\xi^k(P) \mid \xi^k(Q)$	$\xi^k(\mu x.P)$	$=_{\mathrm{df}}$	$\mu x.\xi^k(P)$
$\xi^k(\overline{a}.P)$	$=_{\mathrm{df}}$	$\overline{a}.\xi^\epsilon(P)$						

consider the process $(a{:}0.\mathbf{0} + b{:}2.\mathbf{0}) + c{:}1.\mathbf{0}$ on which no $\mathsf{CCS}^{\mathsf{cw}}$ process is mapped. This example also shows that the notion of compositionality in $\mathsf{CCS}^{\mathsf{cw}}$ is more restrictive than the one in $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$, since a comparable summation can only be extended by summands which have a higher or a lower priority than the already considered summands. The following theorem, which has been proved in [50], makes the semantic relationship between a $\mathsf{CCS}^{\mathsf{cw}}$ process P and its embedding $\xi(P)$ precise.

THEOREM 4.20. *Let $P, Q \in \mathcal{P}^{\mathsf{cw}}$. Then $P \simeq_{cw} Q$ if and only if $\xi(P) \simeq_{ml} \xi(Q)$.*

As a consequence, distributed prioritized strong bisimulation for $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ is also compositional with respect to summation in the sub-calculus of $\mathsf{CCS}^{\mathsf{sl}}_{\mathsf{ml}}$ induced by $\mathsf{CCS}^{\mathsf{cw}}$.

4.10. Concluding Remarks and Related Work. The consideration of a local concept of pre-emption is also made by Hansson and Orava in [35], where Hoare's *Communicating Sequential Processes* (CSP) [40] is extended with priority by assigning natural numbers to actions. As for $\mathsf{CCS}^{\mathsf{sl}}$, they equipped their operational semantics with a notion of location and introduced a sensitivity to locations when defining pre-emption. Indeed, their work served as an inspiration for $\mathsf{CCS}^{\mathsf{sl}}$. However, the authors only conjecture that their version of strong bisimulation is a congruence, and they provide neither an axiomatization for their behavioral relation nor a theory for observational congruence. One may also criticize their semantics as not truly reflecting distributed computation. In particular, despite having a local pre-emptive semantics they compute a global priority for synchronizations.

After stressing the strong similarity of $\mathsf{CCS}^{\mathsf{sl}}$ to the process algebra $\mathsf{CCS}^{\mathsf{cw}}$ in the previous section we focus on the algebraic results established in these frameworks. In [23, 44] the transition relation is directly annotated with pre-emption potentials. By plugging this relation into the definition of standard strong bisimulation one immediately obtains a congruence. In contrast, [27] starts off by defining *naive* distributed prioritized strong bisimulation using the naive transition relation and considers the pre-emption potential subsequently (by introducing the distributed prioritized initial action set condition). Then it is shown that the resulting congruence is the largest congruence in the naive equivalence. Similarly, Jensen [44] defines a naive distributed prioritized weak bisimulation based on the abovementioned annotated transition relation. His naive weak transition relation corresponds to the distributed prioritized weak transition relation in $\mathsf{CCS}^{\mathsf{sl}}$ if the parameter M is dropped. Because of the difference in the naive transition relations our abstraction result is somewhat stronger than Jensen's, although the observational congruences appear to coincide.

One may wonder about the relationship between $\mathsf{CCS}^{\mathsf{sl}}$ and $\mathsf{CCS}^{\mathsf{sg}}$, i.e., the static priority *global* pre-emption language in Section 3. If in $\mathsf{CCS}^{\mathsf{sl}}$ the distributed summation operator is left out and pre-emption is globalized by defining $[m] =_{\mathrm{df}} \mathcal{L}oc$ for all $m \in \mathcal{L}oc$, the operational semantics and the behavioral relations reduce to the corresponding notions presented in Section 3.

Like Camilleri and Winskel, Barrett [7] devises a semantics of occam's priority mechanism that is additionally concerned with fairness aspects. His semantic framework is based on a structural operational semantics augmented with *ready-guard* sets which model possible inputs from the environment. Intuitively, these sets characterize the nature of the contexts in which a transition is possible. Thus, they correspond to the action sets with which the $\mathsf{CCS}^{\mathsf{sl}}$ and the $\mathsf{CCS}^{\mathsf{cw}}$ transition relations are parameterized. However, Barrett is not concerned with investigating behavioral relations but focuses on implementing occam's PRIALT and PRIPAR constructs on the transputer platform.

In addition to Hansson and Orava, other researchers have also extended CSP [40] by a concept of static priority. Inspired by the notion of priority in ADA [47], Fidge [31] has introduced new versions of the operators for external choice, parallel composition by interleaving, and parallel composition by intersection. These favor their left-hand operands similar to the operators investigated by Jensen [44]. The developed semantic theory in [31] is based on *failure semantics* which is made sensitive for local pre-emption. For this purpose, traces are augmented with a *preference function* which identifies the priority relation on the initial action sets of a given process. A related approach has been presented by Lowe [49]. It differs from [31] in that the underlying algebra is a timed version of CSP [29]. Additionally, Lowe aims at obtaining a fully deterministic language by making use of a similar notion of priority as the one proposed by Fidge.

Finally, we remark on the notion of strong and weak bisimulations for $\mathsf{CCS}^{\mathsf{sl}}$. Since the semantic theory reflects local pre-emption, locations are implicitly occurring in our semantic equivalences. However, in contrast to the work on *location equivalences* in [18, 24, 57] we do not consider locations explicitly in our relations. Our objective is not to observe locations but to capture local pre-emption.

5. Dynamic Priority and Global Pre-emption. This section develops a theory in which priorities are dynamic and pre-emption is global. The motivation for this theory originated in a desire to devise a compact model of real-time computation, and we devote significant space to establishing a tight connection between the seemingly different notions of priority and real-time [9]. For this purpose we equip our language with a dynamic priority semantics based on global pre-emption and refer to it as $\mathsf{CCS}^{\mathsf{dg}}$ (CCS with dynamic priority and global pre-emption). The connection with real-time arises when we interpret delays as priorities: the longer the delay preceding an action, the lower is its priority. This approach contrasts significantly with more traditional accounts of real-time, where the only notion of pre-emption arises in the context of the *maximal progress assumption* [13, 74] which states that time may only pass if the system under consideration cannot engage in any further internal computation. The main result of this section is the formalization of a one-to-one correspondence between the strong-bisimulation equivalences induced by dynamic priority and real-time semantics.

Unlike the process algebras with priority considered so far, actions in $\mathsf{CCS}^{\mathsf{dg}}$ do have priority values that may change as systems evolve. Accordingly, we slightly alter our point of view regarding actions and priorities by separating action names from their priority values; that is, an action's priority is no longer implicit in its port name. In this vein, we take the set of actions $\mathcal{A}$ to be $\{\alpha, \beta, \ldots\}$. We also allow priority values to come from the full set $\mathbb{N}$ of natural numbers rather than a finite set. Our syntax of processes will then require that each action is equipped with a priority value taken from $\mathbb{N}$.

The structure of this section is as follows. Section 5.1 briefly presents a real-time semantics of our language, whereas the dynamic priority semantics is introduced in Section 5.2. The one-to-one correspondence is established in Section 5.3. Finally, Section 5.4 contains our concluding remarks and discusses related work.

5.1. Real-time Semantics. We first introduce a real-time semantics for our language, referred to as $\mathsf{CCS}^{\mathsf{rt}}$ semantics, which explicitly represents timing behavior. The semantics of a process is defined by a *labeled transition system* which contains explicit *clock transitions* – each representing a delay of one time unit – as well as *action transitions*. With respect to clock transitions, the operational semantics is set up such that processes willing to communicate with some process running in parallel are able to wait until the communication partner is ready. However, as soon as it is available, the communication has to take place, i.e., further idling is prohibited. This assumption is usually referred to as *maximal progress assumption* [74] or *synchrony hypothesis* [13].

Formally, the labeled transition system corresponding to a process P is a four-tuple $\langle \mathcal{P}, \mathcal{A} \cup \{1\}, \longmapsto, P \rangle$ where $\mathcal{P}$ is the set of states, $\mathcal{A} \cup \{1\}$ the alphabet satisfying $1 \notin \mathcal{A}$, $\longmapsto$ is the transition relation, and P represents the start state. The transition relation $\stackrel{1}{\longmapsto} \subseteq \mathcal{P} \times \mathcal{P}$ for clock transitions is defined in Table 5.1. Regarding action transitions, it coincides with the one for traditional CCS where the Rule (Act) is replaced by the axiom $\alpha{:}0.P \stackrel{\alpha}{\longmapsto} P$. For the sake of simplicity, we use γ as representative of $\mathcal{A} \cup \{1\}$, and write $P \stackrel{\gamma}{\longmapsto} P'$ instead of $\langle P, \gamma, P' \rangle \in \longmapsto$. If $\gamma \in \mathcal{A}$ we speak of an *action transition*, otherwise of a *clock transition*. Sometimes it is convenient to write $P \stackrel{\gamma}{\longmapsto}$ for $\exists P' \in \mathcal{P}.\, P \stackrel{\gamma}{\longmapsto} P'$. In order to ensure maximal progress our operational semantics is set up in a way such that $P \stackrel{1}{\not\longmapsto}$ whenever $P \stackrel{\tau}{\longmapsto}$, i.e., clock transitions are pre-empted as long as P can engage in internal computation.

TABLE 5.1
Operational semantics for $\mathsf{CCS}^{\mathsf{rt}}$

tNil	$\dfrac{-}{\mathbf{0} \stackrel{1}{\longmapsto} \mathbf{0}}$	tRec	$\dfrac{P[\mu x.P/x] \stackrel{1}{\longmapsto} P'}{\mu x.P \stackrel{1}{\longmapsto} P'}$
tAct1	$\dfrac{-}{\alpha{:}k.P \stackrel{1}{\longmapsto} \alpha{:}(k-1).P}\ k > 0$	tAct2	$\dfrac{-}{a{:}0.P \stackrel{1}{\longmapsto} a{:}0.P}$
tSum	$\dfrac{P \stackrel{1}{\longmapsto} P' \quad Q \stackrel{1}{\longmapsto} Q'}{P + Q \stackrel{1}{\longmapsto} P' + Q'}$	tCom	$\dfrac{P \stackrel{1}{\longmapsto} P' \quad Q \stackrel{1}{\longmapsto} Q'}{P \mid Q \stackrel{1}{\longmapsto} P' \mid Q'}\ P \mid Q \stackrel{\tau}{\not\longmapsto}$
tRel	$\dfrac{P \stackrel{1}{\longmapsto} P'}{P[f] \stackrel{1}{\longmapsto} P'[f]}$	tRes	$\dfrac{P \stackrel{1}{\longmapsto} P'}{P \setminus L \stackrel{1}{\longmapsto} P' \setminus L}$

Intuitively, the process $\alpha{:}k.P$, where $k > 0$, may engage in a clock transition and then behave like $\alpha{:}(k-1).P$. The process $\alpha{:}0.P$ performs an α transition to become process P. Moreover, if $\alpha \not\equiv \tau$, it may also idle by executing a clock transition to itself. Time has to proceed equally on both sides of summation, i.e., $P + Q$ can engage in a clock transition and, thus, delay the nondeterministic choice if and only if both P and Q can engage in a clock transition, i.e., time is a deterministic concept. Similar to summation, P and Q have to synchronize on clock transitions according to Rule (tCom). Its side condition implements maximal progress by ensuring that there is no pending communication between P and Q. Although this condition is negative, our semantics is still well-defined [1, 72]. A semantic theory based on the notion of *bisimulation* [52] has been developed for $\mathsf{CCS}^{\mathsf{rt}}$ [55]. For the purposes of this section we restrict ourselves to *strong* temporal bisimulation, a congruence which is defined as follows.

DEFINITION 5.1 (Temporal Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a* temporal bisimulation *if for every $\langle P, Q\rangle \in \mathcal{R}$ and $\gamma \in \mathcal{A} \cup \{1\}$ the following holds: $P \stackrel{\gamma}{\longmapsto} P'$ implies $\exists Q'.\, Q \stackrel{\gamma}{\longmapsto} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$. We write $P \sim_{rt} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some temporal bisimulation $\mathcal{R}$.*

The reader might observe that $\mathsf{CCS}^{\mathsf{rt}}$ semantics unfolds every delay value into a sequence of elementary time units. For example, the process $a{:}k.\mathbf{0}$ has $k+2$ states, namely $\mathbf{0}$ and $a{:}l.\mathbf{0}$ where $0 \leq l \leq k$ (see also Figure 5.1 in Section 5.3). Representing $a{:}k.\mathbf{0}$ by a single transition labeled by $a{:}k$ leading to the state $\mathbf{0}$ would definitely be more efficient. This idea of compacting the state space of real-time systems can be implemented by viewing k as a *priority value* assigned to action a. In other words, one may consider the delay value k as the *time-stamp* of action a [43].

5.2. Dynamic Priority Semantics. In order to make the above intuition precise, we formally introduce $\mathsf{CCS}^{\mathsf{dg}}$, i.e., a dynamic priority semantics for our language. The notion of pre-emption incorporated in $\mathsf{CCS}^{\mathsf{dg}}$ is similar to $\mathsf{CCS}^{\mathsf{sg}}$; it naturally mimics the maximal progress assumption employed in $\mathsf{CCS}^{\mathsf{rt}}$ semantics. Formally, the $\mathsf{CCS}^{\mathsf{dg}}$ semantics of a process P is given by a labeled transition system $\langle \mathcal{P}, \mathcal{A} \times \mathbb{N}, \longrightarrow, P\rangle$. The presentation of the operational rules for the transition relation $\longrightarrow$ requires two auxiliary definitions.

TABLE 5.2
Potential initial action sets for CCS^{dg}

$\mathrm{I}^k(\alpha{:}l.P)$	$=$	$\{\alpha{:}l \mid l = k\}$	$\mathrm{I}^k(P \mid Q)$	$=$	$\mathrm{I}^k(P) \cup \mathrm{I}^k(Q) \cup \{\tau{:}k \mid \mathrm{I}^k(P) \cap \overline{\mathrm{I}^k(Q)} \neq \emptyset\}$
$\mathrm{I}^k(P+Q)$	$=$	$\mathrm{I}^k(P) \cup \mathrm{I}^k(Q)$	$\mathrm{I}^k(P[f])$	$=$	$\{f(\alpha){:}l \mid \alpha{:}l \in \mathrm{I}^k(P)\}$
$\mathrm{I}^k(\mu x.P)$	$=$	$\mathrm{I}^k(P[\mu x.P/x])$	$\mathrm{I}^k(P \setminus L)$	$=$	$\{\alpha{:}l \in \mathrm{I}^k(P) \mid \alpha \notin (L \cup \overline{L})\}$

First, we introduce *potential initial action sets* as defined in Table 5.2, taking account of the actions and their priority values in which a given process can potentially engage. Note that these sets are only supersets of the initial actions of processes since they do not take *pre-emption* into account. However, this is sufficient for our purposes concerning pre-emption since $\tau \notin \mathrm{I}^{<k}(P)$ if and only if $\nexists l < k.\, P \stackrel{\tau:l}{\longrightarrow}$, where $\mathrm{I}^{<k}(P) =_{\mathrm{df}} \bigcup\{\mathrm{I}^l(P) \mid l < k\}$ (cf. Section 3.6).

TABLE 5.3
Priority adjustment function

$[\mathbf{0}]^k$	$=_{\mathrm{df}}$	$\mathbf{0}, \quad [x]^k =_{\mathrm{df}} x$	$[\mu x.P]^k$	$=_{\mathrm{df}}$	$[P[\mu x.P/x]]^k$
$[\alpha{:}l.P]^k$	$=_{\mathrm{df}}$	$\alpha{:}(l-k).P$ if $l > k$	$[\alpha{:}l.P]^k$	$=_{\mathrm{df}}$	$\alpha{:}0.P$ if $l \leq k$
$[P+Q]^k$	$=_{\mathrm{df}}$	$[P]^k + [Q]^k$	$[P \mid Q]^k$	$=_{\mathrm{df}}$	$[P]^k \mid [Q]^k$
$[P[f]]^k$	$=_{\mathrm{df}}$	$[P]^k[f]$	$[P \setminus L]^k$	$=_{\mathrm{df}}$	$[P]^k \setminus L$

As second auxiliary definition for presenting the transition relation, we introduce a *priority adjustment function* as shown in Table 5.3. Intuitively, our semantics is set up in a way such that if one parallel component of a process engages in a transition with priority k, then the priority values of all initial actions at every other parallel component have to be decreased by k, i.e., those actions become equally "more urgent." Thus, the semantics of parallel composition deploys a kind of *fairness assumption*, and priorities

have a *dynamic* character. More precisely, the priority adjustment function applied to a process P and a natural number k, denoted as $[P]^k$, returns a process term which is "identical" to P except that the priority values of the initial, top-level actions are decreased by k. Note that a priority value cannot become less than 0, and the phrase "identical" does not mean syntactic equality but syntactic equality *up to unfolding* of recursion.

TABLE 5.4
Operational semantics for CCS^{dg}

Act1	$\dfrac{-}{a{:}k.P \xrightarrow{a:l} P}\ l \geq k$	Act2	$\dfrac{-}{\tau{:}k.P \xrightarrow{\tau:k} P}$
Sum1	$\dfrac{P \xrightarrow{\alpha:k} P'}{P+Q \xrightarrow{\alpha:k} P'}\ \tau \notin \mathrm{I}^{<k}(Q)$	Sum2	$\dfrac{Q \xrightarrow{\alpha:k} Q'}{P+Q \xrightarrow{\alpha:k} Q'}\ \tau \notin \mathrm{I}^{<k}(P)$
Com1	$\dfrac{P \xrightarrow{\alpha:k} P'}{P\mid Q \xrightarrow{\alpha:k} P'\mid [Q]^k}\ \tau \notin \mathrm{I}^{<k}(P\mid Q)$	Rel	$\dfrac{P \xrightarrow{\alpha:k} P'}{P[f] \xrightarrow{f(\alpha):k} P'[f]}$
Com2	$\dfrac{Q \xrightarrow{\alpha:k} Q'}{P\mid Q \xrightarrow{\alpha:k} [P]^k\mid Q'}\ \tau \notin \mathrm{I}^{<k}(P\mid Q)$	Res	$\dfrac{P \xrightarrow{\alpha:k} P'}{P \setminus L \xrightarrow{\alpha:k} P' \setminus L}\ \alpha \notin L \cup \overline{L}$
Com3	$\dfrac{P \xrightarrow{a:k} P' \quad Q \xrightarrow{\overline{a}:k} Q'}{P\mid Q \xrightarrow{\tau:k} P'\mid Q'}\ \tau \notin \mathrm{I}^{<k}(P\mid Q)$	Rec	$\dfrac{P[\mu x.P/x] \xrightarrow{\alpha:k} P'}{\mu x.P \xrightarrow{\alpha:k} P'}$

The operational rules in Table 5.4 capture the following intuition. The process $a{:}k.P$ may engage in action a with priority value $l \geq k$ yielding process P. The side condition $l \geq k$ reflects that k does not specify an exact priority but the *maximum* priority of the initial transition of $a{:}k.P$. It may also be interpreted as *lower-bound* "timing constraint." Due to the notion of pre-emption incorporated in $\mathsf{CCS}^{\mathsf{dg}}$, $\tau{:}k.P$ may not perform the initial τ-transition with a priority value less than k. The process $P+Q$ may behave like P (Q) if Q (P) does not pre-empt the considered transition by being able to engage in a higher prioritized internal transition. Thus, the notion of global pre-emption reflects implicit *upper-bound* "timing constraints." The process $P \mid Q$ denotes the *parallel composition* of P and Q according to an interleaving semantics with synchronized communication on complementary actions of P and Q having the same priority value k which results in the internal action τ attached with priority value k (cf. Rule (Com3)). The interleaving Rules (Com1) and (Com2) incorporate the dynamic behavior of priority values as explained in the previous paragraph. Their side conditions implement global pre-emption. The semantics for *restriction*, *relabeling*, and *recursion* is straightforward. As for $\mathsf{CCS}^{\mathsf{rt}}$, we may adapt a notion of strong bisimulation, called *prioritized bisimulation*.

DEFINITION 5.2 (Prioritized Bisimulation). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is called* prioritized bisimulation *if for $\langle P, Q\rangle \in \mathcal{R}$, $\alpha \in \mathcal{A}$, and $k \in \mathbb{N}$ the following holds: $P \xrightarrow{\alpha:k} P'$ implies $\exists Q'.\, Q \xrightarrow{\alpha:k} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$. We write $P \sim_{dg} Q$ if there exists a prioritized bisimulation $\mathcal{R}$ such that $\langle P, Q\rangle \in \mathcal{R}$.*

5.3. Relating Dynamic Priority and Real-time Semantics. In this section we show that $\mathsf{CCS}^{\mathsf{dg}}$ and $\mathsf{CCS}^{\mathsf{rt}}$ semantics are closely related. The underlying intuition is best illustrated by a simple example dealing with the prefixing operator. Figure 5.1 depicts the dynamic priority and real-time semantics of the process $a{:}k.\mathbf{0}$. Both transition systems intuitively reflect that the process $a{:}k.\mathbf{0}$ must at least delay k time units before it may engage in an a-transition. According to $\mathsf{CCS}^{\mathsf{rt}}$ semantics this process consecutively engages in k clock transitions passing the states $a{:}(k-i).\mathbf{0}$, for $0 \leq i \leq k$, before it may either continue idling in state $a{:}0.\mathbf{0}$ or perform an a-transition to the inaction process $\mathbf{0}$. Thus, time is explicitly part of states and made visible by clock transitions, each representing a step consuming one time unit. In contrast, the dynamic priority semantics encodes the delay of at least k time units in the transitions rather than in the states. Hence, it possesses only the two states $a{:}k.\mathbf{0}$ and $\mathbf{0}$ connected via transitions labeled by $a{:}l$ for $l \geq k$. Although at first sight it seems that the price for saving intermediate states is to be forced to deal with infinite-branching, an upper bound of l can be given. In our example this upper bound is k itself, since a delay by more than k time units only results in idling and does not enable new or disable existing system behavior. Therefore, the dynamic priority transition system of $a{:}k.\mathbf{0}$ just consists of the two states $a{:}k.\mathbf{0}$ and $\mathbf{0}$ and a *symbolic* transition labeled by $a{:}k$, whereas the real-time transition system has $k+2$ states and $k+2$ transitions. The following proposition formally states that $\mathsf{CCS}^{\mathsf{dg}}$ semantics can indeed be understood as an efficient encoding of $\mathsf{CCS}^{\mathsf{rt}}$ semantics.

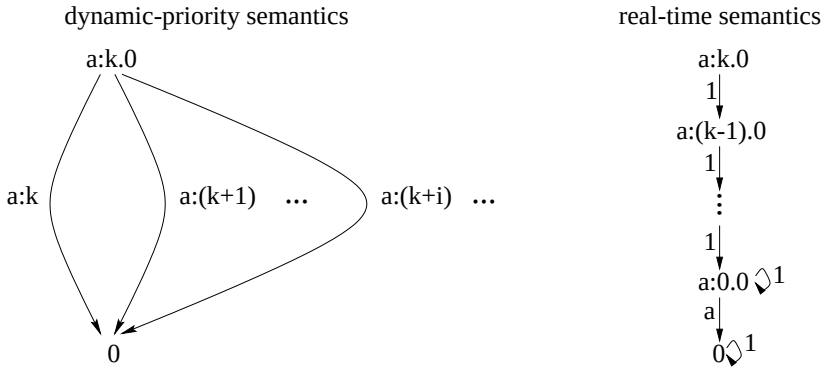

FIG. 5.1. *Relating CCS^{dg} and CCS^{rt} semantics*

PROPOSITION 5.3. *Let $P, P' \in \mathcal{P}$, $\alpha \in \mathcal{A}$, and $k \in \mathbb{N}$. Then*

$$P \xrightarrow{\alpha:k} P' \text{ if and only if } \exists P'' \in \mathcal{P}.\, P \stackrel{1}{\longmapsto}{}^{k} P'' \stackrel{\alpha}{\longmapsto} P' .$$

Proposition 5.3 is the key to prove the main result of this section.

THEOREM 5.4. *Let $P, Q \in \mathcal{P}$. Then $P \sim_{dg} Q$ if and only if $P \sim_{rt} Q$.*

Consequently, prioritized and temporal bisimulation possess the same properties; especially, prioritized bisimulation is a congruence for $\mathsf{CCS}^{\mathsf{dg}}$. Again, proof details can be found in [50].

5.4. Concluding Remarks and Related Work. As shown above, real-time semantics can be encoded by dynamic priority semantics. Moreover, the state spaces of $\mathsf{CCS}^{\mathsf{dg}}$ models are much smaller and the size of the transition relation is at least not worse, but in practice often better, than the one of corresponding $\mathsf{CCS}^{\mathsf{rt}}$ models. This has been demonstrated by formally modeling and verifying several aspects of the widely-used *SCSI-2 bus-protocol* [16], for which the state space of the dynamic priority model is almost an order of magnitude smaller than the one resulting from traditional real-time semantics.

Regarding related work, a similar approach has been made by Jeffrey [43] who has established a formal relationship between a quantitative real-time process algebra and a process algebra with *static* priority which is very similar to $\mathsf{CCS}^{\mathsf{sg}}$ presented in Section 3. Jeffrey also translates real-time to priority based on the idea of time-stamping. In contrast to $\mathsf{CCS}^{\mathsf{rt}}$ semantics, however, a process modeled in Jeffrey's framework may either *immediately* engage in an action transition or idle forever. This semantics does not obey a characteristic of the behavior of *reactive* systems, namely that a process should wait until a communication partner becomes available, instead of engaging in a "livelock." It is only because of this assumption that Jeffrey does not need to choose a *dynamic* priority framework.

In [21] a variant of CCSR [22] has been introduced which allows for modeling not only static priority but also dynamic priority. The main focus of CCSR involves the specification and verification of real-time concurrent systems, including scheduling behavior. Thus, a notion of dynamic priority, such as occurs in *priority-inheritance* and *earliest-deadline-first* scheduling algorithms, is crucial. In [21] dynamic priorities are given as a function of the *history* of the system under consideration, and the operational semantics of CCSR is re-defined to include the historical context. The authors show that dynamic priorities do, in general, not lead to a compositional semantics and give a sufficient condition that ensures compositionality.

6. Priority in Other Process-algebraic Frameworks. This section completes the discussion of related work by focusing on approaches to priority which (i) do either not fit in our classification scheme presented in Section 1, such as approaches for ACP [4], SCCS [68], and *stochastic* [11, 39] or *probabilistic* [71, 45, 69] process algebras, or (ii) are concerned with process-algebraic descriptions of non-process-algebraic languages, such as *Esterel* [12, 13] and *Statecharts* [36, 70].

Baeten, Bergstra, and Klop were the first researchers who investigated priorities in process algebras [4] by developing a notion of priority for the *Algebra of Communicating Processes* (ACP) [8] – a process algebra which is equipped with an axiomatic semantics. Their work is inspired by the insight that it is essential to incorporate an interrupt mechanism in process-algebraic frameworks in order to enhance their expressive power as specification and verification formalisms for concurrent systems. Therefore, a piece of syntax together with semantics defining equations is introduced in [4]. Based on a given partial order $<$ on actions a unary operator θ is defined. Intuitively, $\theta(P)$ is the context of P in which action a has precedence over action b whenever $b < a$, i.e., non-deterministic choices between actions a and b are resolved within $\theta(P)$. Technically, the axiomatic semantics of the new language, notated as a *term rewrite system*, is shown to possess nice algebraic properties such as *confluence* and *termination*. The utility of the theory is demonstrated by simple examples dealing with interrupts, timeouts, and other aspects of real-time behavior. The approach in [4] differs from most other work presented in [10] in that the partial order expressing priorities is fixed with respect to the system under consideration, i.e., the same priority relation holds at all states of the system. For example, if $a < b$ at some state of the system, then $a > b$ cannot be valid at another state, i.e., priorities in [4] are not *globally dynamic* in the sense of [68]. It should also be mentioned that the version of ACP used in [4] does not include a designated internal action, cf. action τ in CCS; a fact which simplifies the development of algebraic theories.

Stochastic process algebras [11, 39], which enhance the expressiveness of classical process algebras by integrating *performance* descriptions of concurrent systems, also define notions of priority. One example of a well-known stochastic process algebra is the *Extended Markovian Process Algebra* (EMPA) [11] whose semantics is given in terms of strong bisimulation, and its static priority approach is adapted from $\mathsf{CCS}^{\mathsf{sg}}$.

Smolka and Steffen [68] have introduced static priority to the *Synchronous Calculus of Communicating Systems* (SCCS) [52] by extending a probabilistic version of this language, known as PCCS [71], whose semantics is given in terms of *probabilistic bisimulation*. Their work shows that the concept of priority is not only related to real-time, as investigated in Section 5, but also to probability. The main idea in [68] is to allow probability guards of value 0 to be associated with alternatives of a probabilistic summation expression. Such alternatives can be chosen only if the non-zero alternatives are precluded by contextual constraints. Thus, priority may be viewed as an extreme case of probability. Most remarkably, the semantics developed in [68] does not employ a notion of pre-emption as one would expect from any priority setting. A conjecture – which if true would justify this situation – is that the very powerful hiding operator of SCCS may destroy the congruence property of bisimulation in the presence of pre-emption.

Tofts has investigated another extension of SCCS, the *Weighted Synchronous Calculus of Communicating Systems* (WSCCS) [69]. Its semantics relies upon a notion of *relative frequency* which is suitable for specifying and reasoning about aspects of priority, probability, and time in concurrent systems. In this approach priority is encoded by means of higher ordinals; a transition has priority over another if their weights are separated at least by a factor of ω. An operator similar to the θ-operator in [4] is defined which extracts the highest priority transitions enabled at a process state by referring to a global notion of pre-emption. In contrast to [4], Toft's operator allows for different priority structures at different states. This concept of priority yields a simpler operational semantics than the one in [68]. For WSCCS, a congruence adapted from strong bisimulation together with an equational characterization, which is sound and complete for finite processes, has been developed.

The concept of pre-emption has also been studied in other synchronous languages, most notably by Berry [12]. His technical framework is based on *Esterel's zero-delay process calculus*, a theoretical version of the Esterel synchronous programming language [13]. The calculus' semantics interprets processes as deterministic mappings from input sequences to output sequences which obey maximal progress [74]. Berry emphasizes the importance of pre-emption in control-dominated reactive and real-time programming. He suggests pre-emption operators to be considered as first-class operators which are fully orthogonal with respect to all other primitives such as concurrency and communication. This is in contrast to the approach chosen for this article in which pre-emption is implicitly encoded as side conditions of operational rules involving nondeterminism. Several examples of useful pre-emption operators are presented and axiomatized in [12], all of which are based on the ideas of *abortion* and *suspension*.

The specification language *Statecharts* [36], for which process-algebraic descriptions of Statecharts' semantics have been developed [70], extends communicating finite automata by concepts of *hierarchy* and *priority*. In Statecharts static priorities can be expressed via the absence of actions, also called *events*, by permitting negated actions as guards, which are referred to as *triggers*. As an example, consider the following term describing a simple statechart: $a : b.\, P + \neg b : c.\, Q$. This term consists of a nondeterministic choice between a b-transition with guard a to process P, and a c-transition with guard $\neg b$ to Q. Intuitively, the statechart may only engage in the latter transition if it cannot execute the former since this one produces the event b which falsifies the guard of the c-transition. Thus, the b-transition is given precedence over the c-transition. In the following we argue that approaches to priority via negated events (cf. [34]) do not go well along with the concept of *hiding* which is used in many process algebras and also in a very popular variant of Statecharts, called ARGOS [51]. Hiding enables one to relabel a visible action into a distinguished invisible action (cf. the internal action τ in CCS). The problem with hiding arises when several events are hidden, i.e.,

all of them are relabeled to the same event and, thus, have the same implicit priority value attached to them. Hence, hiding may destroy priority structures. However, in most other priority approaches considered in this paper priorities are assigned to transitions, thereby allowing for a more fine-granular priority mechanism and avoiding the above-mentioned problem.

7. Conclusions and Directions for Future Work. This article has investigated various aspects of priority in process algebras. The utility of introducing priority to traditional process algebras is to enhance their expressiveness and, thereby, making them more attractive to system designers.

7.1. Conclusions. We have illustrated the most important aspects of priority by defining a prototypic language which extends Milner's Calculus of Communicating Systems (CCS). This language has been equipped with several semantics according to whether priorities are static or dynamic and whether the adapted notion of pre-emption is global or local.

In practice it is easy to determine when to use a static priority and when to use a dynamic priority semantics: for modeling interrupts and prioritized choice constructs a static notion of priority is adequate, whereas for modeling real-time or scheduling behavior dynamic priorities should be considered. However, static priority approaches may also allow for the description of a few, very simple scheduling algorithms, as has been shown in [44] in the presence of a prioritized parallel composition operator. In addition to the dynamic priority approach's ability to express more general scheduling algorithms, it also leads to a more efficient verification of real-time systems since the sizes of system models with respect to dynamic priority semantics are often several orders of magnitude smaller than the ones regarding real-time semantics [16]. If one needs to deal with both interrupt and real-time aspects at the same time, static and dynamic priority approaches must be combined. In this situation each action should be assigned two priority values, the first interpreted as a global priority value for scheduling purposes and the second interpreted as a local priority value for modeling interrupts, where the first priority value has more weight than the second one.

Suitable guidelines supporting the decision in favor of a global or a local notion of pre-emption are the following. A semantics obeying global pre-emption is required when modeling interrupts and prioritized-choice constructs in concurrent, centralized systems or when specifying real-time and scheduling aspects. Global pre-emption also allows for making executions of action sequences atomic. This can be necessary for modeling systems accurately and, as a desired side effect, keeps system models small, thereby enhancing the efficiency of verification procedures [28]. However, when dealing with interrupts or prioritized-choice constructs within distributed systems the concept of global pre-emption is inadequate. Here, the use of local pre-emption does not only lead to an intuitive but also to an implementable semantics since it does not require any knowledge about computations which are internal to other, potentially unknown sites (cf. [26]).

Technically, the three different calculi presented in Sections 3–5 have been equipped with a bisimulation-based semantics. The re-development of the semantic theory of CCS for the static priority calculi included: (i) characterizations of the largest congruences contained in the naive adaptations of the standard strong and weak bisimulations, (ii) encodings of the new behavioral relations as standard strong bisimulations on enriched transition relations, and (iii) axiomatic characterizations of the prioritized strong bisimulations for finite processes. For the dynamic priority calculus strong bisimulation has been served as a semantic tool for establishing a one-to-one correspondence between dynamic priority and real-time semantics. Finally, observe that our semantic theories show that extensions of process algebras by priority do not need to sacrifice the simplicity and the elegance that have made traditional process-algebraic approaches successful.

This article has also surveyed related approaches to priority which are concerned with different process-algebraic calculi. We have classified them according to whether priorities are considered to be static or dynamic and whether their concept of pre-emption is global or local. The concept of priority has also been investigated in other concurrent frameworks, most notably in *Petri Nets* [14, 67]. In this setting priorities are either expressed explicitly by priority relations over transitions [15] or implicitly via *inhibitor arcs* [42]. Finally, it should be mentioned that priorities can *implicitly* arise when studying causality for *mobile processes* (see e.g. [30]). In these approaches, priorities cut off superfluous paths that only present new temporal but not causal dependencies of systems.

7.2. Future Work. In addition to the fact that a calculus combining dynamic priority and local pre-emption has not been developed, yet, also the semantic theories for $\mathsf{CCS}^{\mathsf{sg}}$ and $\mathsf{CCS}^{\mathsf{sl}}$ need to be completed by axiomatizing their *observational* congruences. For *finite* processes, one should be able to establish these axiomatizations using standard techniques [53]. However, for *regular* processes, i.e., the class of finite-state processes not containing recursion through static operators, it is not clear how to obtain completeness. The point is that existing methods for proving completeness of axiomatizations with respect to observational congruences rely on the possibility to remove or to insert τ-cycles in processes [53]. In the context of pre-emption, however, this would possibly change the pre-emption potential of processes and is, thus, semantically incompatible with the prioritized observational congruences presented here. Recently, a similar problem has been attacked in [38] for a stochastic timed process calculus with maximal progress. The definition of observational equivalence employed in that paper differs from Milner's original one by adding a notion of fairness which is sensitive to escaping divergence, i.e., infinite internal computation. However, the authors conjecture that their technique can be adapted to priority frameworks, too.

Most process algebras which have been equipped with a notion of priority rely on an interleaving semantics, handshake communication, and a semantic theory based on bisimulation. It should be investigated in which sense the presented approaches and results, especially regarding local pre-emption, can be adapted to broadcasting calculi such as Hoare's CSP [40]. Moreover, since for semantics based on local pre-emption the usual interleaving law is not valid, it is worth pursuing local pre-emption for non-interleaving semantic frameworks [3, 73]. Preliminary considerations have been made in Jensen's thesis [44]. However, the insights obtained by Jensen are restricted to a structural operational semantics for a CCS-based calculus which is defined using *asynchronous transition systems* [73]. Jensen's results do not comprise a behavioral relation such as bisimulation (cf. [57]). Finally, we want to note that – to the best of our knowledge – extensions of higher-order process algebras [54, 62] with concepts of priority do not yet exist. Thus, it would be interesting to see if some of the presented approaches can be carried over straightforwardly or if any semantic difficulties regarding pre-emption arise.

8. Sources and Acknowledgments. Major parts of this article have been adapted from several publications by the authors which include two Ph.D. theses: the results of Section 3 are taken from [25, 50, 58, 59] and the ones of Section 4 from [27, 50]; Section 5 heavily borrows from material contained in [16, 50]. The authors would like to thank Girish Bhat, Matthew Hennessy, Michael Mendler, and Bernhard Steffen for many discussions about priority in process algebras.

REFERENCES

[1] L. Aceto, W. Fokkink, and C. Verhoef, *Structural operational semantics*, in Bergstra et al. [10]. To appear.

[2] J. Baeten, ed., *Applications of Process Algebra*, Vol. 17 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, England, 1990.

[3] J. Baeten and T. Basten, *Non-interleaving ACP and its application to Petri nets*, in Bergstra et al. [10]. To appear.

[4] J. Baeten, J. Bergstra, and J. Klop, *Syntax and defining equations for an interrupt mechanism in process algebra*, Fundamenta Informaticae IX, (1986), pp. 127–168.

[5] J. Baeten and J. Klop, eds., *CONCUR '90 (Concurrency Theory)*, Vol. 458 of Lecture Notes in Computer Science, Amsterdam, August 1990, Springer-Verlag.

[6] J. Baeten and W. Weijland, *Process Algebra*, Vol. 18 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, England, 1990.

[7] G. Barrett, *The semantics of priority and fairness in occam*, in Proceedings of Mathematical Foundations of Programming Semantics, M. Main, A. Melton, M. Mislove, and D. Schmidt, eds., Vol. 442 of Lecture Notes in Computer Science, Springer-Verlag, 1989, pp. 194–208.

[8] J. Bergstra and J. Klop, *Algebra of communicating processes with abstraction*, Theoretical Computer Science, 37 (1985), pp. 77–121.

[9] J. Bergstra, C. Middelburg, and Y. Usenko, *Discrete time process algebra and the semantics of SDL*, in Bergstra et al. [10]. To appear.

[10] J. Bergstra, A. Ponse, and S. Smolka, eds., *Handbook of Process Algebra*, Elsevier, 1999. To appear.

[11] M. Bernardo and R. Gorrieri, *Extended markovian process algebra*, in CONCUR '96 (Concurrency Theory), U. Montanari and V. Sassone, eds., Vol. 1119 of Lecture Notes in Computer Science, Pisa, Italy, August 1996, Springer-Verlag, pp. 315–330.

[12] G. Berry, *Preemption in concurrent systems*, in Foundations of Software Technology and Theoretical Computer Science, Vol. 761 of Lecture Notes in Computer Science, Springer-Verlag, 1993, pp. 72–93.

[13] G. Berry and G. Gonthier, *The ESTEREL synchronous programming language, design, semantics, implementation*, Science of Computer Programming, 19 (1992), pp. 87–152.

[14] E. Best, R. Devillers, and M. Koutny, *A consistent model for nets and process algebras*, in Bergstra et al. [10]. To appear.

[15] E. Best and M. Koutny, *Petri net semantics of priority systems*, Theoretical Computer Science, 96 (1992), pp. 175–215.

[16] G. Bhat, R. Cleaveland, and G. Lüttgen, *A practical approach to implementing real-time semantics*, Annals of Software Engineering, 7 (1999). Special issue on Real-time Software Engineering.

[17] T. Bolognesi and E. Brinksma, *Introduction to the ISO specification language LOTOS*, Computer Networks and ISDN Systems, 14 (1987), pp. 25–59.

[18] G. Boudol, I. Castellani, M. Hennessy, and A. Kiehn, *Observing localities*, Theoretical Computer Science, 114 (1993), pp. 31–61.

[19] J. Bradfield and C. Stirling, *Modal process logics*, in Bergstra et al. [10]. To appear.

[20] P. Brémond-Grégoire, J.-Y. Choi, and I. Lee, *A complete axiomatization of finite-state ACSR processes*, Information and Computation, 138 (1997), pp. 124–159.

[21] P. Brémond-Grégoire, S. Davidson, and I. Lee, *CCSR92: Calculus for communicating shared resources with dynamic priorities*, in Purushothaman and Zwarico [66], pp. 65–85.

[22] P. Brémonde-Grégoire, I. Lee, and R. Gerber, *A process algebra of communicating shared resources with dense time and priorities*, Theoretical Computer Science, 189 (1997), pp. 179–219.

[23] J. Camilleri and G. Winskel, *CCS with priority choice*, Information and Computation, 116 (1995), pp. 26–37.

[24] I. Castellani, *Locations and distributed processes*, in Bergstra et al. [10]. To appear.

[25] R. Cleaveland and M. Hennessy, *Priorities in process algebras*, Information and Computation, 87 (1990), pp. 58–77.

[26] R. Cleaveland, G. Lüttgen, and M. Mendler, *An algebraic theory of multiple clocks*, in CONCUR '97 (Concurrency Theory), A. Mazurkiewicz and J. Winkowski, eds., Vol. 1243 of Lecture Notes in Computer Science, Warsaw, Poland, July 1997, Springer-Verlag, pp. 166–180.

[27] R. Cleaveland, G. Lüttgen, and V. Natarajan, *A process algebra with distributed priorities*, Theoretical Computer Science, 195 (1998), pp. 227–258.

[28] R. Cleaveland, V. Natarajan, S. Sims, and G. Lüttgen, *Modeling and verifying distributed systems using priorities: A case study*, Software–Concepts and Tools, 17 (1996), pp. 50–62.

[29] J. Davies and S. Schneider, *A brief history of Timed CSP*, Theoretical Computer Science, 138 (1995), pp. 243–271.

[30] P. Degano and C. Priami, *Causality of mobile processes*, in International Conference on Automata, Languages and Programming (ICALP '95), Z. Fülöp and F. Gécseg, eds., Vol. 944 of Lecture Notes in Computer Science, Szeged, Hungary, July 1995, Springer-Verlag, pp. 660–671.

[31] C. Fidge, *A formal definition of priority in CSP*, ACM Transactions on Programming Languages and Systems, 15 (1993), pp. 681–705.

[32] R. Gerber and I. Lee, *CCSR: A calculus for communicating shared resources*, in Baeten and Klop [5], pp. 263–277.

[33] ——, *A resourced-based prioritized bisimulation for real-time systems*, Information and Computation, 113 (1994), pp. 102–142.

[34] S. German, *Programming in a general model of synchronization*, in CONCUR '92 (Concurrency Theory), R. Cleaveland, ed., Vol. 630 of Lecture Notes in Computer Science, Stony Brook, New York, August 1992, Springer-Verlag, pp. 534–549.

[35] H. Hansson and F. Orava, *A process calculus with incomparable priorities*, in Purushothaman and Zwarico [66], pp. 43–64.

[36] D. Harel, *Statecharts: A visual formalism for complex systems*, Science of Computer Programming, 8 (1987), pp. 231–274.

[37] M. Hennessy, *Algebraic Theory of Processes*, MIT Press, Boston, 1988.

[38] H. Hermanns and M. Lohrey, *Priority and maximal progress are completely axiomatisable*, in CONCUR '98 (Concurrency Theory), D. Sangiorgi and R. de Simone, eds., Vol. 1466 of Lecture Notes in Computer Science, Nice, France, September 1998, Springer-Verlag.

[39] H. Hermanns, M. Rettelbach, and T. Weiss, *Formal characterisation of immediate actions in spa with nondeterministic branching*, The Computer Journal, 38 (1995), pp. 530–541.

[40] C. Hoare, *Communicating Sequential Processes*, Prentice-Hall, London, 1985.

[41] INMOS Limited, *Occam Programming Manual*, International Series in Computer Science, Prentice Hall, 1984.

[42] R. Janicki and M. Koutny, *Semantics of inhibitor nets*, Information and Computation, 123 (1995), pp. 1–17.

[43] A. Jeffrey, *Translating timed process algebra into prioritized process algebra*, in Proceedings of Symposium on Real-Time and Fault-Tolerant Systems (FTRTFT '92), J. Vytopil, ed., Vol. 571 of Lecture Notes in Computer Science, Nijmegen, The Netherlands, January 1992, Springer-Verlag, pp. 493–506.

[44] C.-T. Jensen, *Prioritized and Independent Actions in Distributed Computer Systems*, Ph.D. thesis, Aarhus University, Denmark, August 1994.

[45] B. Jonsson, K. Larsen, and W. Yi, *Probabilistic extensions of process algebra*, in Bergstra et al. [10]. To appear.

[46] P. Kanellakis and S. Smolka, *CCS expressions, finite state processes, and three problems of equivalence*, Information and Computation, 86 (1990), pp. 43–68.

[47] Kempe Software Capital Enterprises, *Ada95 reference manual: Language and standard libraries*, 1995. Available at http://www.adahome.com.

[48] L. Lamport, *What it means for a concurrent program to satisfy a specification: Why no one has specified priority*, in Twelfth Annual ACM Symposium on Principles of Programming Languages (POPL '85), Orlando, Florida, January 1985, IEEE Computer Society Press, pp. 78–83.

[49] G. Lowe, *Probabilistic and prioritized models of timed CSP*, Theoretical Computer Science, 138 (1995), pp. 315–352.

[50] G. Lüttgen, *Pre-emptive Modeling of Concurrent and Distributed Systems*, Ph.D. thesis, University of Passau, Germany, May 1998. Published by Shaker Verlag, Aachen, Germany.

[51] F. Maraninchi, *The ARGOS language: Graphical representation of automata and description of reactive systems*, in IEEE Workshop on Visual Languages, October 1991.

[52] R. Milner, *Communication and Concurrency*, Prentice-Hall, London, 1989.

[53] ——, *A complete axiomatization for observational congruence of finite-state behaviours*, Information and Computation, 81 (1989), pp. 227–247.

[54] R. Milner, J. Parrow, and D. Walker, *A calculus of mobile processes, part I and II*, Information and Computation, 100 (1992), pp. 1–77.

[55] F. Moller and C. Tofts, *A temporal calculus of communicating systems*, in Baeten and Klop [5], pp. 401–415.

[56] U. Montanari and D. Yankelevich, *Location equivalence in a parametric setting*, Theoretical Computer Science, 149 (1995), pp. 299–332.

[57] M. Mukund and M. Nielsen, *CCS, locations and asynchronous transition systems*, in Foundations of Software Technology and Theoretical Computer Science (FSTTCS '92), R. Shyamasundar, ed., Vol. 652 of Lecture Notes in Computer Science, Springer-Verlag, 1992, pp. 328–341.

[58] V. Natarajan, *Degrees of Delay: Semantic Theories for Priority, Efficiency, Fairness, and Predictability in Process Algebras*, Ph.D. thesis, North Carolina State University, August 1996.

[59] V. Natarajan, L. Christoff, I. Christoff, and R. Cleaveland, *Priorities and abstraction in process algebra*, in Foundations of Software Technology and Theoretical Computer Science (FSTTCS '94), P. Thiagarajan, ed., Vol. 880 of Lecture Notes in Computer Science, Madras, India, December 1994, Springer-Verlag, pp. 217–230.

[60] R. Paige and R. Tarjan, *Three partition refinement algorithms*, SIAM Journal of Computing, 16 (1987), pp. 973–989.

[61] D. Park, *Concurrency and automata on infinite sequences*, in Proceedings of 5th G.I. Conference on Theoretical Computer Science, P. Deussen, ed., Vol. 104 of Lecture Notes in Computer Science, Springer-Verlag, 1981, pp. 167–183.

[62] J. Parrow, *Mobility in process algebras*, in Bergstra et al. [10]. To appear.

[63] G. Plotkin, *A structural approach to operational semantics*, Tech. Report DAIMI-FN-19, Computer Science Department, Aarhus University, Denmark, 1981.

[64] K. Prasad, *Programming with broadcasts*, in CONCUR '93 (Concurrency Theory), E. Best, ed., Vol. 715 of Lecture Notes in Computer Science, Hildesheim, Germany, August 1993, Springer-Verlag, pp. 173–187.

[65] ——, *Broadcasting with priority*, in Proceedings of the 5th European Symposium on Programming, Vol. 788 of Lecture Notes in Computer Science, Edinburgh, U.K., April 1994, Springer-Verlag, pp. 469–484.

[66] P. Purushothaman and A. Zwarico, eds., *First North American Process Algebra Workshop*, Workshops in Computing, Stony Brook, New York, August 1992, Springer-Verlag.

[67] W. Reisig, *Petri Nets: An Introduction*, Springer-Verlag, 1985.

[68] S. Smolka and B. Steffen, *Priority as extremal probability*, Formal Aspects of Computing, 8 (1996), pp. 585–606.

[69] C. Tofts, *Processes with probablities, priority and time*, Formal Aspects of Computing, 6 (1994), pp. 536–564.

[70] A. Uselton and S. Smolka, *A compositional semantics for statecharts using labeled transition systems*, in CONCUR '94 (Concurrency Theory), B. Jonsson and J. Parrow, eds., Vol. 836 of Lecture Notes in Computer Science, Uppsala, Sweden, August 1994, Springer-Verlag, pp. 2–17.

[71] R. van Glabbeek, S. Smolka, and B. Steffen, *Reactive, generative, and stratified models of probabilistic processes*, Information and Computation, 121 (1995), pp. 59–80.

[72] C. Verhoef, *A congruence theorem for structured operational semantics with predicates and negative premises*, Nordic Journal of Computing, 2 (1995), pp. 274–302.

[73] G. Winskel and M. Nielsen, *Models for concurrency*, in Handbook of Logic Computer Science, S. Abramsky, D. Gabbay, and T. Maibaum, eds., Vol. 4, Oxford Science Publications, 1995, pp. 1–148.

[74] W. Yi, *CCS + time = an interleaving model for real time systems*, in International Conference on Automata, Languages and Programming (ICALP '91), J. L. Albert, B. Monien, and M. R. Artalejo, eds., Vol. 510 of Lecture Notes in Computer Science, Madrid, July 1991, Springer-Verlag, pp. 217–228.